

**Universidad  
Rey Juan Carlos**

Escuela Técnica Superior  
de Ingeniería Informática

**Grado en Diseño y Desarrollo de Videojuegos**

**Curso 2019-2023**

**Trabajo Fin de Grado**

**PERSONALIZACIÓN DE AVATARES HUMANOS  
MEDIANTE MODELOS DE DEEP LEARNING**

**Autor: Javier Serrano del Amo**

**Tutor: Jorge Félix López Moreno**

**Co-Tutor: Dan Casas Guix**

**Fecha: 15/07/2023**



# Agradecimientos

Quiero expresar mi más profunda gratitud a todas las personas que han contribuido a la realización de este Trabajo de Fin de Grado.

En primer lugar, deseo agradecer a mi tutor de TFG, Jorge, por su orientación y dedicación que ha volcado en mi, que junto con Dan y Álex me han guiado, enseñado y aconsejado en todo el proceso, aprendiendo mucho en el camino.

Además, quiero expresar mi agradecimiento a mis amigos, quienes me han acompañado durante esta etapa universitaria, haciendo que la vida sea un poco más sencilla y divertida.

Por último, no puedo estar más agradecido por tener la familia que tengo. Siempre me brindan un apoyo incondicional en todo lo que hago, aunque no lo comprendan.

Muchas gracias a todos.



# Resumen

Este trabajo se centra en la creación de avatares humanos 3D utilizando modelos paramétricos basados en datos de humanos reales (SMPL-X y DECA), con el objetivo de lograr avatares realistas a partir de datos de entrada de fácil acceso como pueden ser las imágenes tomadas con un teléfono móvil. Para lograrlo, se busca la unión de dichos modelos paramétricos y abordan los desafíos asociados, incluyendo varias mejoras. Además, se hace especial énfasis en el proceso de texturización de la cabeza, incluso corrigiendo las posibles imprecisiones de alineación en las texturas presentes en algunos avatares debido a la baja resolución de las imágenes de entrada.

## Palabras clave:

- SMPL
- SMPL-X
- DECA
- FLAME
- PCA
- *Inpainting*
- *MetaHuman*
- SEDDI
- CAESAR *dataset*
- *Offset*
- GUI
- *Landmarks*
- Renderizador inverso



# Índice de contenidos

<b>Índice de figuras</b>	<b>XI</b>
<b>1. Introducción</b>	<b>1</b>
1.1. Contexto y alcance . . . . .	2
<b>2. Trabajos previos</b>	<b>4</b>
<b>3. Implementación</b>	<b>8</b>
3.1. Punto de unión entre modelos . . . . .	10
3.2. Tratamiento de geometría . . . . .	12
3.3. Tratamiento de texturas . . . . .	29
<b>4. Resultados</b>	<b>53</b>
4.1. Ejecución del modelo . . . . .	53
4.2. Alineación de ojos . . . . .	64
4.3. <i>Training</i> y optimización . . . . .	66
<b>5. Conclusiones y trabajos futuros</b>	<b>70</b>
<b>6. Bibliografía</b>	<b>73</b>



# Índice de figuras

3.1. Imágenes de entrada de DECA para la identidad del sujeto. . . . .	9
3.2. Imagen de entrada de DECA para la expresión del sujeto. . . . .	9
3.3. Modelo SMPL-X femenino con parámetros neutros. . . . .	10
3.4. Modelo DECA. . . . .	11
3.5. Unión de modelos mediante correspondencias entre vértices. . . . .	11
3.6. Cuerpo generado mediante la suma de <i>offsets</i> entre cabezas. . . . .	12
3.7. Diferencias entre modelos en la zona de la boca. . . . .	13
3.8. Corrección de boca en modelo SMPL-X. . . . .	14
3.9. Deformación de cabeza por modelo posado. . . . .	15
3.10. Corrección de la deformación de cabeza con modelo posado. . . . .	15
3.11. Alineación con referencia al primer vértice del cuello. . . . .	16
3.12. Alineación con referencia al centro de gravedad. . . . .	16
3.13. Modificación de cabeza a causa de parámetros de cuerpo SMPL-X. .	17
3.14. Deformación de cabeza con parámetros poco realistas de SMPL-X. .	18
3.15. Corrección de cabeza a causa de parámetros SMPL-X. . . . .	19
3.16. Preservación de indentidad con parámetros poco realistas. . . . .	19
3.17. Modelo sin suavizado de clavícula/cuello. . . . .	20
3.18. Modelo con suavizado de clavícula/cuello. . . . .	21
3.19. <i>Add-on</i> de Blender para la aplicación de pesos sobre los vértices. .	21
3.20. Comparación de cuerpo neutro y cuerpo generado mediante optimizador. . . . .	23
3.21. Comparación de cuerpo neutro y cuerpo generado mediante optimizador con cabeza escalada. . . . .	23
3.22. Cuerpo SMPL-X con cabeza posada. . . . .	24
3.23. Modelo SMPL-X eliminando información de pose y de expresión residual. . . . .	25
3.24. Inferencia de cuerpo con cabeza posada y cabeza no posada. . . . .	25
3.25. Creación de cuerpo y expresión SMPL-X mediante interfaz gráfica de usuario. . . . .	27
3.26. <i>Pipeline</i> de geometría. . . . .	28
3.27. Comparación de cabeza DECA con y sin mapa de albedo. Vista frontal. . . . .	29

3.28. Comparación de cabeza DECA con y sin mapa de albedo. Vista trasera. . . . .	29
3.29. Comparación de cabeza DECA con y sin mapa de normales. Modelo con expresión neutra. . . . .	30
3.30. Comparación de cabeza DECA con y sin mapa de normales. Modelo con expresión personalizada. . . . .	30
3.31. Comparación de cabeza DECA con y sin mapa de normales y con mapa de albedo. . . . .	31
3.32. Textura mal proyectada. . . . .	32
3.33. Textura bien proyectada. . . . .	33
3.34. Textura de albedo generada por DECA. . . . .	33
3.35. Textura de albedo generada por FLame2SMPLX. . . . .	34
3.36. Texturizado SMPL-X mediante FLame2SMPLX. . . . .	34
3.37. Textura de albedo genérica de SMPL-X utilizada como plantilla. .	35
3.38. Combinación de texturas de albedo SMPL-X y DECA. . . . .	36
3.39. Combinación de texturas de mapa de normales de SMPL-X y DECA.	36
3.40. Textura DECA con zona oscura en la zona de la frente. . . . .	37
3.41. Modelo SMPL-X texturizado con zona oscura en la zona de la frente.	38
3.42. Selección manual de área oscura. . . . .	38
3.43. Textura de DECA con relleno de píxeles en la zona de la frente. .	39
3.44. SMPL-X con relleno de píxeles en la zona de la frente. . . . .	40
3.45. Efecto de costura en texturas generadas por DECA. . . . .	41
3.46. Modelo DECA con textura sin efecto costura. . . . .	41
3.47. Modelos DECA con texturas suavizadas en la zona trasera. . . . .	42
3.48. Proceso de unión de vértices en la zona de la clavícula para corrección de textura. . . . .	43
3.49. Imagen generada por el decodificador de DECA. . . . .	44
3.50. Imagen generada por el renderizador de DECA que guarda la información del sombreado. . . . .	45
3.51. Albedo generado mediante textura decodificada y sombreado de renderizador. . . . .	45
3.52. Textura <i>ground truth</i> generada por el renderizador de DECA. .	46
3.53. Máscara predeterminada de DECA. . . . .	46
3.54. Textura de albedo generada por DECA para utilizar sobre el modelo.	46
3.55. Máscara personalizada para corrección de ojos. . . . .	47
3.56. Textura de albedo generada por DECA utilizando la máscara personalizada. . . . .	47
3.57. Modelo DECA usando como albedo la textura generada con la máscara personalizada. . . . .	48
3.58. Ojos de la textura de albedo generada por DECA para distintos sujetos. Píxel verde: centro ideal. Píxel azul: centro deseado. . . . .	49
3.59. Desplazamiento de textura para corrección de ojos. . . . .	49

---

3.60. <i>Dataset</i> utilizada para el entrenamiento de red neuronal. . . . .	50
3.61. <i>Arquitectura de red neuronal seleccionada para el alineamiento de ojos. En verde la arquitectura basada en capas convolucionales. En naranja la arquitectura MLP completamente conectada.</i> . . . . .	51
3.62. Resultado del uso de corrección de ojos mediante red neuronal. . . . .	52
4.1. Resultados de texturizado y expresión en sujeto número 1. . . . .	54
4.2. Resultados de topología en sujeto número 1. . . . .	55
4.3. Resultados de texturizado y expresión en sujeto número 2. . . . .	56
4.4. Resultados de topología en sujeto número 2. . . . .	57
4.5. Resultados de texturizado y expresión en sujeto número 3. . . . .	58
4.6. Resultados de topología en sujeto número 3. . . . .	59
4.7. Resultados de texturizado y expresión en sujeto número 4. . . . .	60
4.8. Resultados de topología en sujeto número 4. . . . .	61
4.9. Resultados de texturizado y expresión en sujeto número 5. . . . .	62
4.10. Resultados de topología en sujeto número 5. . . . .	63
4.11. Resultados sobre varios sujetos en la utilización de la herramienta de alineación de ojos. . . . .	64
4.12. Resultados sobre varios sujetos en la utilización de la herramienta de alineación de ojos. . . . .	65
4.13. Función de pérdida en el proceso de inferencia de cuerpo. Ejemplo 1. .	66
4.14. Función de pérdida en el proceso de inferencia de cuerpo. Ejemplo 2. .	66
4.15. Función de pérdida sobre conjuntos de datos de entrenamiento para red neuronal alineadora de ojos. . . . .	67
4.16. Función de pérdida sobre conjuntos de datos de validación para red neuronal alineadora de ojos. . . . .	67
4.17. Función de pérdida en el proceso de alineación de cabezas. . . . .	68

# 1

## Introducción

La creación de avatares humanos realistas siempre ha suscitado gran interés en el mundo digital, ya sea en la industria del videojuego, animación, moda, o cualquier otro. Sin embargo, la creación de dichos avatares es un proceso lento, ya que involucra la creación de una malla topológicamente aceptable para el contexto que le ocupa y su texturizado.

Este proceso manual conlleva un gasto de recursos que, dados los últimos avances tecnológicos y el surgimiento de nuevos métodos de aprendizaje profundo, deben ser reducidos y automatizados cada vez más.

Es por ello que este trabajo se centra en la automatización de dicho proceso, explorando diferentes modelos basados en aprendizaje profundo, con el objetivo de crear avatares humanos realistas de una manera sencilla para cualquiera que lo use, listo para ser usado en un entorno virtual.

De esta manera, este trabajo utiliza dos modelos pre-existentes, siendo SMPL-X (SMPL eXpressive)<sup>[26]</sup>, que es una extensión del modelo SMPL (Skinned Multi-Person Linear Model)<sup>[24]</sup>, y DECA (Detailed Expression Capture and Animation)<sup>[19]</sup> que hacen un gran trabajo en dos aspectos completamente distintos, y son combinados en un mismo modelo que resalta sus capacidades.

Por lo tanto, al final del proceso se obtendrá un avatar humano 3D parcialmente texturizado, que puede ser posado y que contará con una expresión facial si se desea.

## 1.1. Contexto y alcance

En los últimos años, la generación de avatares 3D realistas ha adquirido una gran relevancia en diversos campos, como la industria del entretenimiento, los videojuegos, la realidad virtual y el mundo de la moda. La capacidad de crear representaciones digitales precisas de personas reales tiene aplicaciones significativas en la animación, la simulación y otras áreas afines. Estas necesidades se alinean con una de las líneas de investigación de la empresa SEDDI[10], así como con el proyecto de investigación V+REAL (Plan Nacional) en el grupo de investigación MSLab, la cual pretende crear avatares humanos realistas con la mínima entrada de datos posible.

En la actualidad, este campo ha experimentado un notable crecimiento y sus efectos se pueden observar en diversos trabajos, como en los MetaHuman[7] de Unreal Engine[11]. Aunque estos trabajos demuestran resultados prometedores, existen otros modelos que exploran y potencian las posibles vulnerabilidades, creando modelos anatómicamente más versátiles en términos de forma y expresión, con un control más detallado y una mayor adaptabilidad a diferentes aplicaciones, entre otras mejoras.

En este contexto, los modelos paramétricos de vanguardia DECA y SMPL-X han surgido como herramientas prometedoras para generar avatares humanos altamente realistas. DECA se especializa en la captura y animación de expresiones faciales detalladas, mientras que SMPL-X ofrece una representación completa del cuerpo humano, incluyendo la forma y pose corporal.

Este trabajo se centra en el desarrollo de un flujo de trabajo que permitirá combinar y utilizar de manera conjunta los modelos DECA y SMPL-X para generar avatares 3D realistas de múltiples sujetos simultáneamente, a la par que se amplian sus funcionalidades. Se prestará especial atención a diversos problemas de alineación de texturas y el tratamiento de estas en la parte de la cabeza, sin hacer ningún tratamiento respecto a las texturas del cuerpo.

Además, se creará una interfaz de usuario que permita seleccionar el cuerpo deseado basado en 10 parámetros, así como una expresión facial adicional personalizada. Por otro lado, se incluirá una herramienta de generación de cuerpo automático basado en la forma de la cabeza generada por DECA, en caso de que el usuario no quiera diseñar un cuerpo mediante la interfaz mencionada. De esta manera, al final del pipeline se obtendrán varios modelos preparados para su integración en un entorno digital.



# 2

## Trabajos previos

A continuación, se presenta un resumen de los trabajos previos relevantes en el campo de estudio:

### **Blend Skinning**

Método utilizado en el mundo de la animación que se basa en la unión de la superficie de un modelo tridimensional en una estructura esquelética subyacente. Consiste en transformar cada vértice de la malla utilizando una combinación ponderada de las transformaciones de los huesos cercanos, según los pesos de influencia asignados. La técnica más común es el Linear Blend Skinning (LBS)[22], que utiliza una transformación lineal.

### **Blend Shapes**

Técnica utilizada para controlar y modificar la forma de un modelo tridimensional de manera específica y controlada. Consiste en crear una serie de formas predefinidas que representan expresiones faciales, poses o deformaciones deseadas. Estas formas son generadas a partir de la forma base del modelo. Se pueden generar formas como por ejemplo la sonrisa de un sujeto, y ser posteriormente combinada con el uso de *Blend Skinning* para posar el modelo.

### Large Scale 3D Morphable Models

Modelo que consiste en la creación de modelos morfológicos 3D a gran escala utilizando una gran cantidad de datos de formas faciales. Utilizan diversas técnicas de adquisición de datos como la fotogrametría para capturar una gran variedad de rostros en diferentes poses y expresiones. A partir de estos datos, desarrollan un modelo estadístico que captura las variaciones y correlaciones entre las formas faciales, permitiendo posteriormente generar nuevas formas de cara realistas.

#### Basel Face Models[20]

Modelo que tiene el objetivo de capturar y representar de manera detallada la variabilidad de las formas faciales humanas. Combina información geométrica y textural para representar las formas de la cara. El modelo proporciona una descripción paramétrica de las formas faciales, lo que significa que se pueden ajustar los parámetros del modelo para generar una variedad de formas faciales personalizadas.

#### SMPL

El modelo SMPL (Skinned Multi-Person Linear Model) es ampliamente conocido en la comunidad de visión por computadora debido a su capacidad para representar de manera precisa y eficiente la forma del cuerpo humano. Fue entrenado utilizando un conjunto de datos de alrededor de 2000 escáneres humanos de ambos géneros (*CAESAR dataset*)[28], lo que le permite representar una amplia variedad de cuerpos mediante un espacio latente de tan solo 10 parámetros, gracias al uso de la técnica de PCA[27] que permite reducir la dimensionalidad de los parámetros más expresivos, convirtiendo en una tarea sencilla la generación de cuerpos. Dichos cuerpos generados cuentan con una geometría de 6890 vértices.

## SMPL-X

En este trabajo se usará SMPL-X (SMPL eXpressive), siendo este una extensión del modelo SMPL, que cuenta con una serie de parámetros adicionales que le permite captar otros detalles anatómicos, como pueden ser expresiones faciales o formas de pies y manos. Aunque estos parámetros adicionales no tienen relevancia en este trabajo, el uso de SMPL-X está justificado ya que integra el modelo paramétrico FLAME en la zona de la cabeza, el cual será el punto de unión entre los modelos DECA y SMPL-X, ya que DECA también se basa en el modelo de FLAME. Debido a esto, la geometría del modelo SMPL-X aumenta respecto al del modelo base en las zonas de los pies, manos y cabeza, sumando un total de 10475 vértices.

## FLAME

FLAME (Faces Learned with an Articulated Model and Expressions)[23] es un modelo paramétrico para la creación de cabezas humanas y sus respectivos rasgos faciales y expresiones, que se basa en un espacio lineal entrenado utilizando una amplia colección de 3800 escáneres. Esta metodología permite al modelo crear una amplia variedad de cabezas humanas, capturando de forma precisa la forma, pose corporal y expresiones faciales detalladas. La geometría generada por FLAME es compatible con la cabeza del modelo SMPL-X, y consta de un total de 5023 vértices.

## DECA

DECA (Detailed Expression Capture and Animation) es un modelo que permite la generación de cabezas a partir de imágenes de entrada de tamaño 224x224 píxeles, teniendo como base FLAME, el cual le permite eliminar la necesidad de especificar los parámetros de FLAME de forma explícita. DECA cuenta con la capacidad de distinguir entre los rasgos faciales que definen la identidad y aquellos que definen la expresión. De esta manera, al proporcionar una segunda imagen que capture la expresión facial deseada, es posible generar una malla adicional que conserve la identidad del sujeto y refleje la expresión reflejada en dicha imagen. Al estar basado en FLAME, la malla generada por DECA consta de 5023 vértices. Por otro lado, es capaz de generar mapas de texturas de albedo y de normales, por lo que al final del proceso se obtendrá una malla totalmente texturizada.



# 3

## Implementación

En este apartado se presenta una descripción detallada del trabajo realizado, abordando los problemas encontrados y las soluciones o enfoques planteados hasta llegar al producto final. Dado que este trabajo se basa principalmente en las investigaciones previas de DECA y SMPL, se explicarán las características implementadas siguiendo un orden cronológico en cuanto a su implementación y se agruparán según su relación con los trabajos previos mencionados. En consecuencia, se dedicarán secciones específicas a las implementaciones relacionadas con la geometría del cuerpo, centradas en el trabajo de SMPL-X y, en menor medida, DECA. Además, se abordarán las técnicas de tratamiento de texturas, enfocándose en el trabajo realizado por DECA. Estas secciones son de vital importancia en este apartado, ya que brindan una visión clara y completa de las contribuciones realizadas en ambos aspectos.

Por otro lado, en esta sección se muestra una gran variedad de imágenes de ejemplo de modelos construidos en distintas fases del desarrollo, tanto de SMPL-X como de DECA. Teniendo en cuenta que DECA trabaja con imágenes de entrada, los ejemplos mostrados han sido contruidos mediante las imágenes de entrada de la Figura 3.1 para la identidad del sujeto, y la imagen de la Figura 3.2 para la expresión.

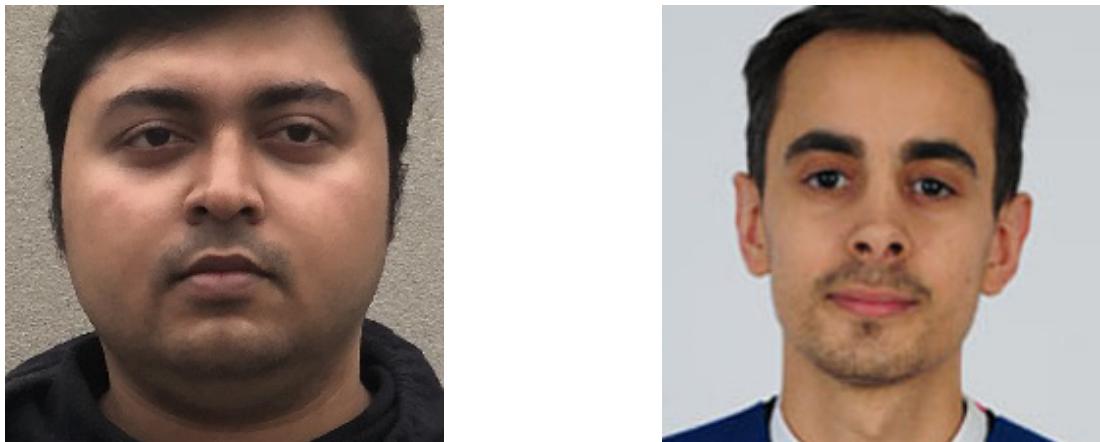


Figura 3.1: Imágenes de entrada de DECA para la identidad del sujeto.



Figura 3.2: Imagen de entrada de DECA para la expresión del sujeto.

### 3.1. Punto de unión entre modelos

El primer desafío que se plantea es la fusión de ambos modelos debido a sus diferencias topológicas. SMPL-X cuenta con una topología de 10.475 vértices, mientras que DECA tiene 5.023 vértices. Afortunadamente, tanto SMPL-X como DECA utilizan FLAME como modelo para la cabeza, lo cual simplifica el proceso de conexión al centrarse únicamente en el mapeo entre los vértices correspondientes.

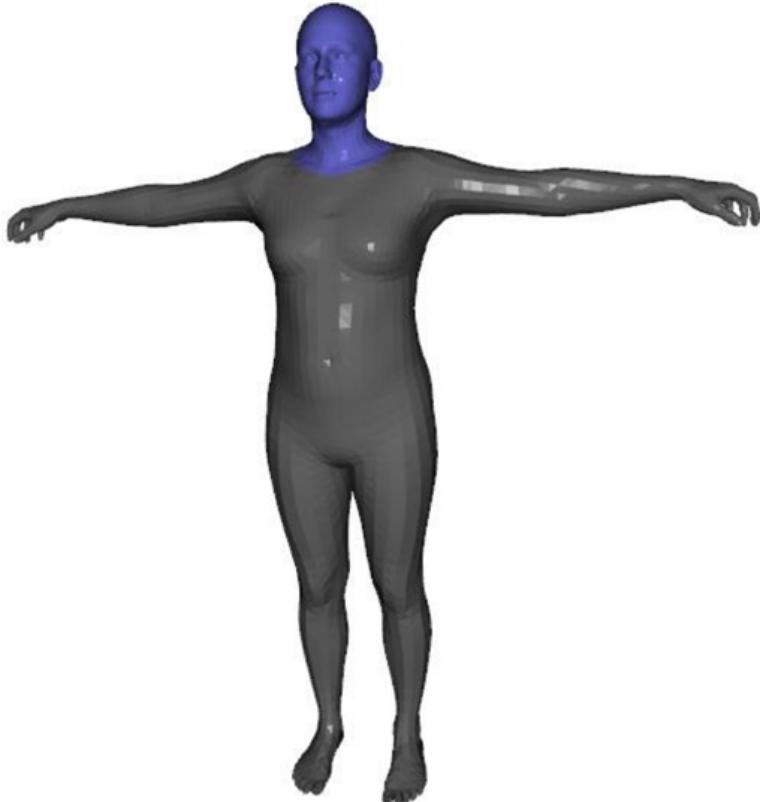
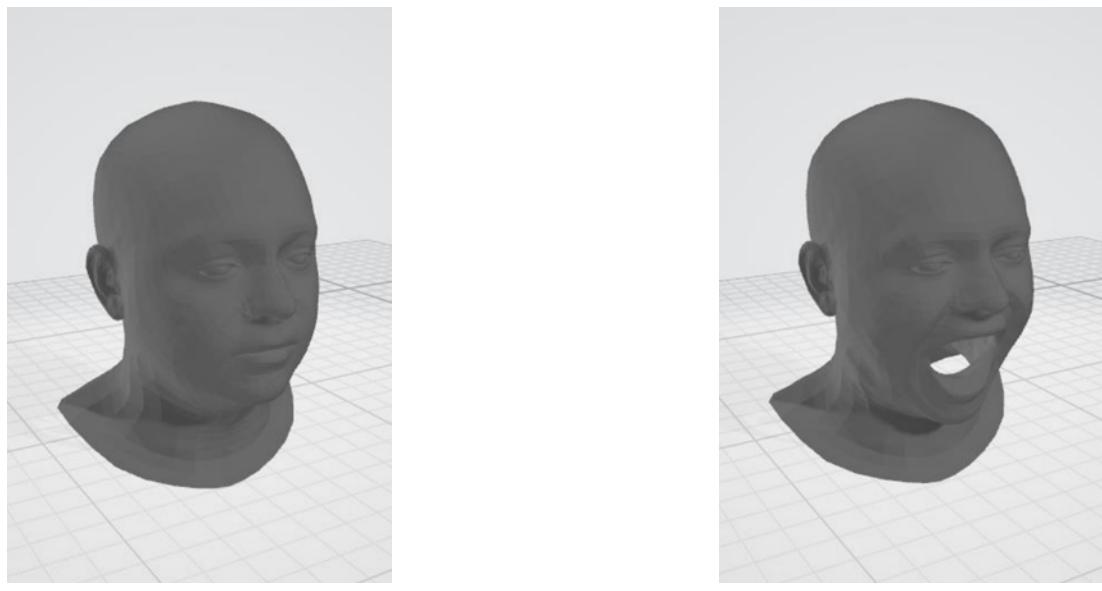


Figura 3.3: Modelo SMPL-X femenino con parámetros neutros.

En las primeras versiones, estas correspondencias no estaban disponibles, pero ahora se pueden encontrar en el siguiente [enlace](#)[2]. Con esta información, la fusión de los dos modelos se realiza de manera directa.



(a) Modelo sin expresión

(b) Modelo con expresión

Figura 3.4: Modelo DECA.

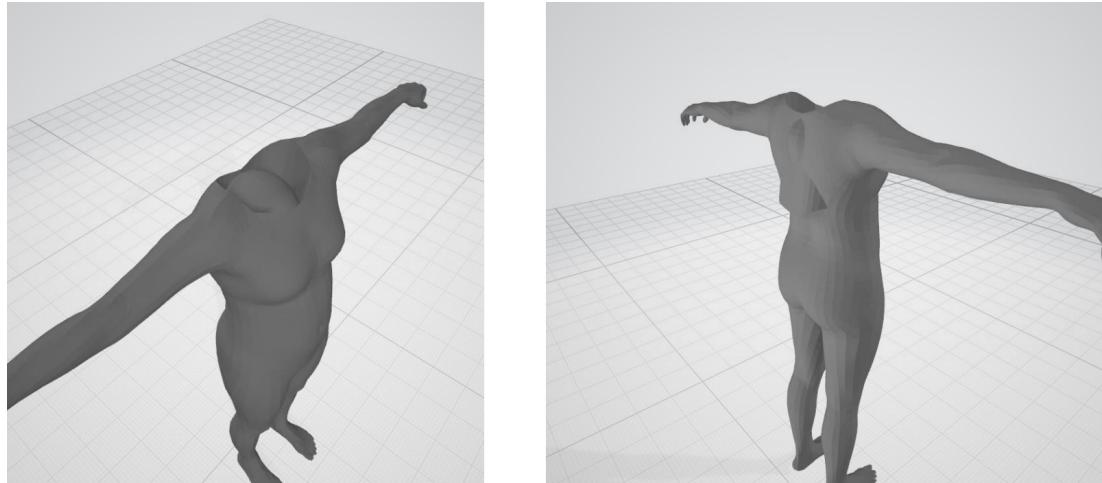


Figura 3.5: Unión de modelos mediante correspondencias entre vértices.

La elección de SMPL-X sobre SMPL se justifica por su uso de FLAME para la cabeza. SMPL base es más limitado en este aspecto, con menos geometría en las manos, pies y cara, y un total de 6.890 vértices.

## 3.2. Tratamiento de geometría

### Transformación y alineación de cabezas

Como se ha observado, la sustitución directa de los vértices no coloca la cabeza en una posición adecuada, ya que termina dentro del tronco. Por lo tanto, la premisa en la que se trabaja es en encontrar los *offsets* entre ambas cabezas, refiriéndose estos a las diferencias o desplazamientos entre los vértices correspondientes de ambas mallas. Dado que ambas se construyen sobre FLAME, el modelo neutral (con los 10 parámetros establecidos en cero) debe coincidir en ambos casos.

En consecuencia, se utiliza un modelo neutro de FLAME, disponible [aquí](#)[3], que se compara con la cabeza generada por DECA. Debido a que ambos modelos carecen de cuerpo, se instancian con una alineación inicial adecuada, ya que sus puntos de pivote coinciden, y se determinan sus *offsets* mediante la ecuación 3.1. Dado que se están generando cuerpos SMPL-X neutros hasta ahora, se añaden los desplazamientos a la región de la cabeza, transformando la identidad de la cabeza neutra a la generada mediante DECA, consiguiendo el cuerpo de la Figura 3.6.

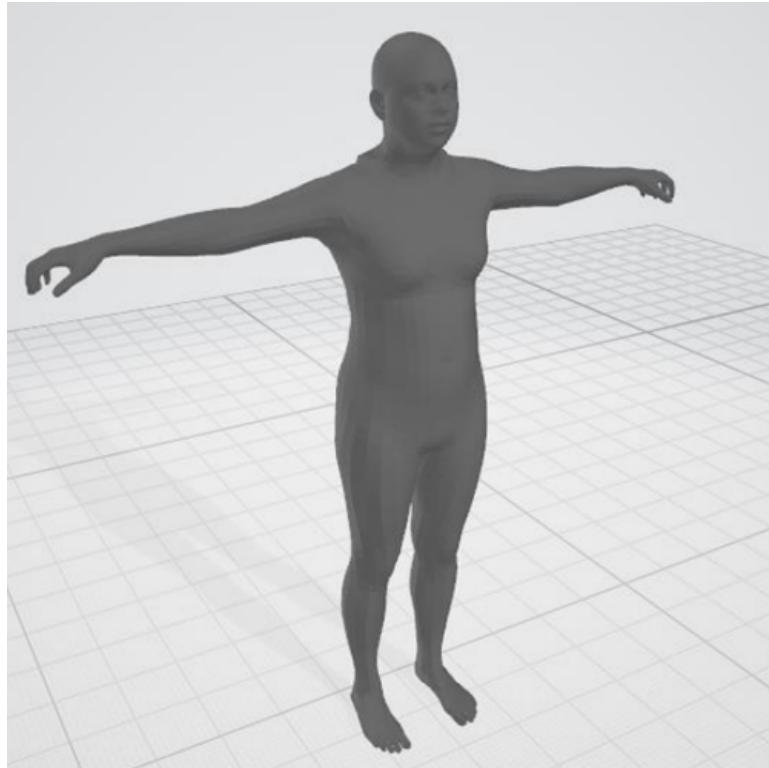
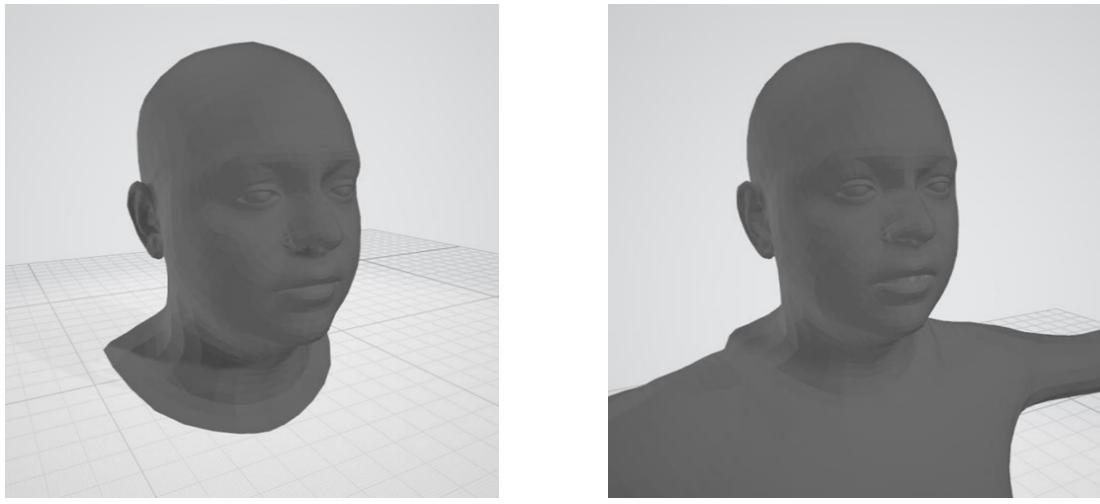


Figura 3.6: Cuerpo generado mediante la suma de *offsets* entre cabezas.

$$O(m_{\text{malla } 1}, m_{\text{malla } 2}) = \{v_i | v_i = v_i^{\text{malla } 2} - v_i^{\text{malla } 1}, \forall i = 1, 2, \dots, 5023\} \quad (3.1)$$

En la Figura 3.7 se observa una discrepancia en la región de la boca entre dos modelos aparentemente idénticos, lo que resulta en una ligera apertura de boca en el modelo de SMPL-X después de aplicar los *offsets* anteriormente calculados. Debido a que ambos modelos se basan en FLAME, se asumió que los modelos neutros serían idénticos. Sin embargo, se ha descubierto que este desajuste en la boca es inherente al modelo neutro predeterminado de SMPL-X.



(a) Modelo DECA deseado.

(b) Modelo SMPL-X con desfase.

Figura 3.7: Diferencias entre modelos en la zona de la boca.

Para abordar esta cuestión, se requiere realizar un *offset* adicional entre otro par de cabezas, en concreto los modelos neutros de FLAME y SMPL-X, mediante la ecuación 3.1.

Para llevar a cabo este proceso, es crucial alinear adecuadamente la geometría de estas cabezas en primer lugar, ya que difieren en su posición de instanciación. El enfoque de alineación seleccionado, aunque no es óptimo y se abordará más adelante en el proyecto, utiliza como referencia única el primer vértice del cuello, que es el más cercano al cuerpo. En un principio, se asumió erróneamente que todos los modelos generados presentaban cuellos idénticos. Una vez que las cabezas están alineadas, se determinan los desplazamientos y se incorporan al modelo SMPL-X, logrando un modelo sin discrepancias en la zona de la boca, como se puede apreciar en la Figura 3.8. Hasta este momento la cabeza es modificada mediante dos *offsets* distintos, representados en la Ecuación 3.2.

$$O_p = O(m_{\text{flame neutro}}, m_{\text{deca}}) + O(m_{\text{smplx neutro}}, m_{\text{flame neutro}}) \quad (3.2)$$



Figura 3.8: Corrección de boca en modelo SMPL-X.

### Corrección en el posado del modelo SMPL-X

Cuando se crea un modelo utilizando SMPL-X, existe la opción de asignarle una pose inicial. Esta funcionalidad es efectiva cuando no se realizan modificaciones al modelo, a diferencia de lo que se lleva a cabo en este caso con la cabeza. Como consecuencia, la cabeza puede experimentar deformaciones según la pose del modelo, ya que los desplazamientos entre los pares de cabezas se calculan posteriormente. Aunque estas deformaciones podrían parecer significativas si se las imagina mentalmente, en realidad generan modelos totalmente plausibles, lo que dificulta la detección de este error. Al trabajar con el modelo que se lleva viendo anteriormente, al aplicarle una pose arbitraria se puede observar la deformación de la cabeza en la Figura 3.9.



Figura 3.9: Deformación de cabeza por modelo posado.

Afortunadamente, SMPL-X permite el posado de cuerpos ya creados de una manera sencilla. Por lo tanto, el cuerpo inicial ahora será instanciado en una pose neutra, se le aplicarán las modificaciones deseadas y, finalmente, se posará en la pose deseada como se muestra en la Figura 3.10.



Figura 3.10: Corrección de la deformación de cabeza con modelo posado.

### Mejora en la alineación de cabezas

Inicialmente, se asumió que la región del cuello era idéntica en ambos modelos de cabeza, por lo que se utilizó el primer vértice de cada modelo, ubicado al inicio del cuello, como referencia para poder alinearlas. Aunque este enfoque ha mostrado resultados favorables, se reconoce que no es el más apropiado.

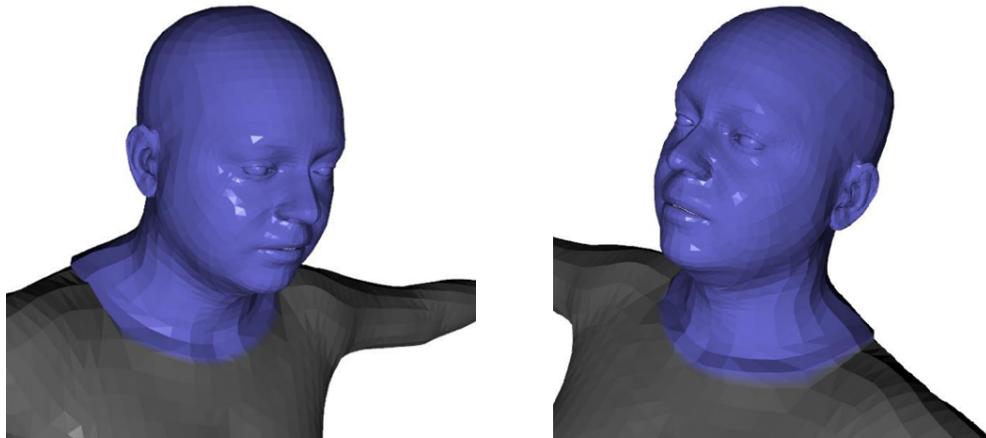


Figura 3.11: Alineación con referencia al primer vértice del cuello.

Por tanto, a partir de este punto se procede a determinar los centros de gravedad de ambas cabezas y se calculan los *offsets* en base a esta información. Esta nueva aproximación permite observar cómo la cabeza se desplaza ligeramente hacia una posición más centrada en comparación con la configuración anterior.

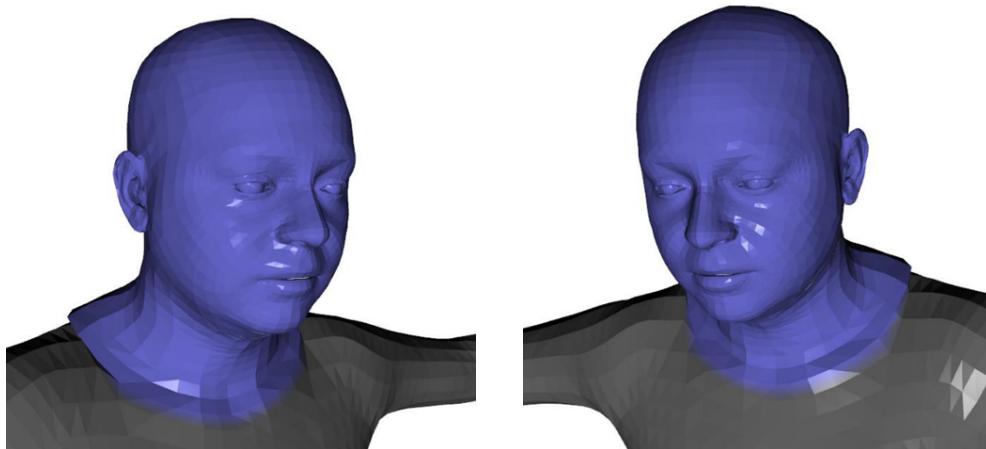


Figura 3.12: Alineación con referencia al centro de gravedad.

### Modificación de parámetros de cuerpo

Hasta el momento, se ha trabajado con el cuerpo neutro de SMPL-X, lo cual carece de sentido seguir prolongando, ya que se desea abordar la generación de cuerpos personalizados. Para ello, es necesario realizar modificaciones en los parámetros de SMPL-X. Sin embargo, surge un problema al ajustar los parámetros para obtener el cuerpo deseado, ya que estos también afectan a la forma de la cabeza, lo que impide obtener la cabeza deseada al sumarle los *offsets* calculados anteriormente, ya que estos se tenían en cuenta respecto a la cabeza neutra.

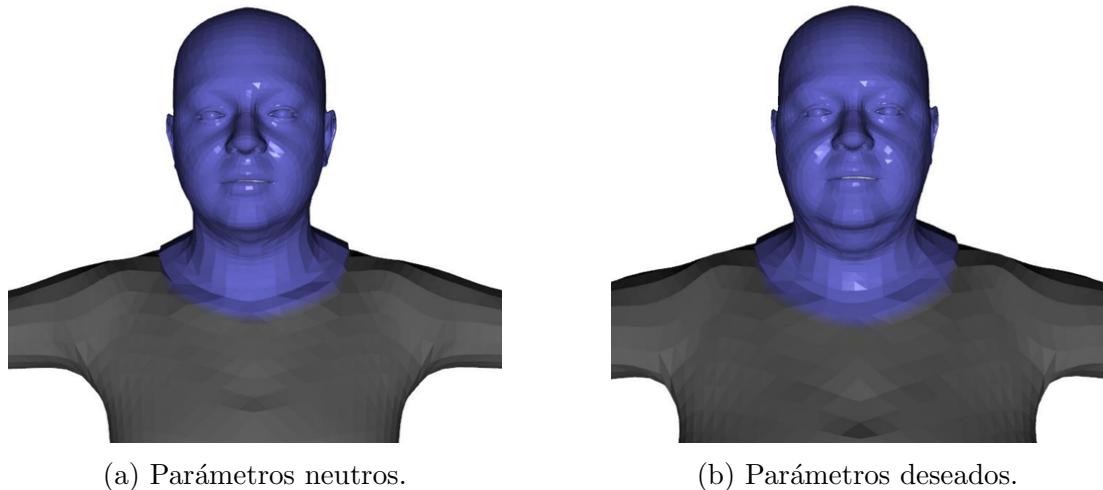


Figura 3.13: Modificación de cabeza a causa de parámetros de cuerpo SMPL-X.

Este cambio se vuelve aún más evidente cuando se prueban ciertos parámetros que generan un cuerpo extremadamente poco realista, como refleja la Figura 3.14.

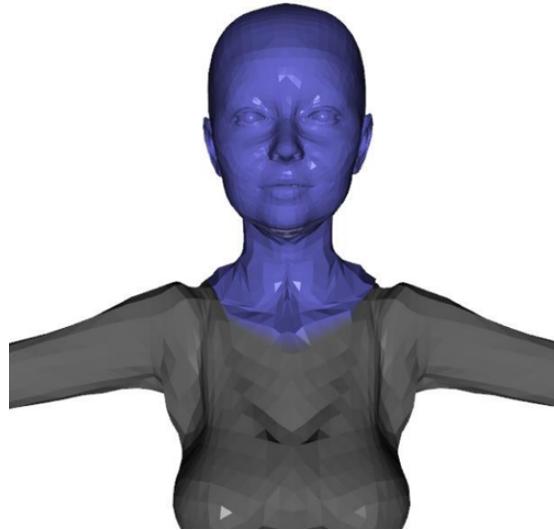


Figura 3.14: Deformación de cabeza con parámetros poco realistas de SMPL-X.

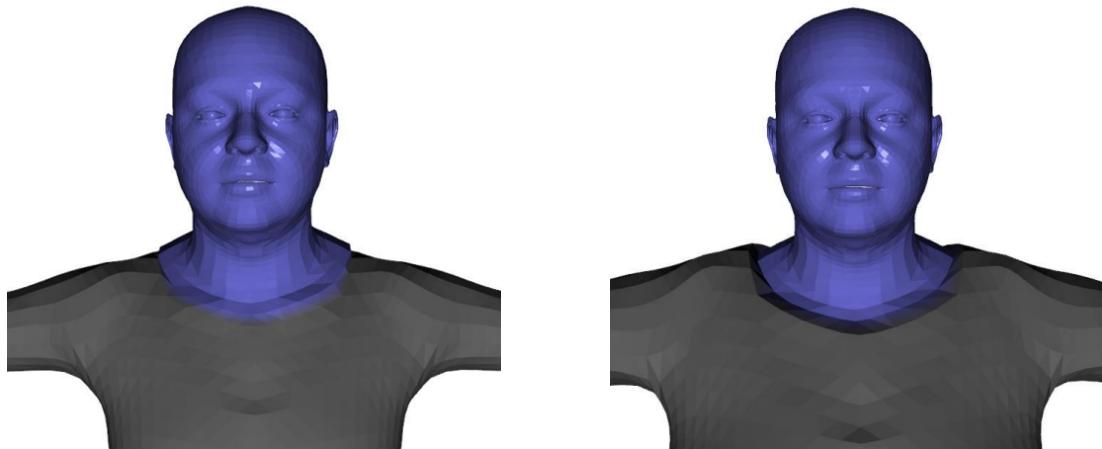
Por lo tanto, se vuelve imprescindible realizar un nuevo cálculo de *offset* entre otros pares de cabezas usando la Ecuación 3.1, siendo estas la generada por SMPL-X al modificar sus parámetros y la versión neutra, que pueden ser observadas en la Figura 3.13. Una vez se añaden estos *offsets*, la identidad inicial se preserva para cualquier tipo de cuerpo, incluso con el muy poco realista que se puede observar en la Figura 3.16.

En este punto, la cabeza generada por SMPL-X se ve transformada mediante 3 *offsets* de pares de cabezas distintos representados en la Ecuación 3.3, y se mantendrá así en todo el trabajo para modelos sin expresión.

$$\begin{aligned} O_{\text{NE}} = & O(m_{\text{flame neutro}}, m_{\text{deca}}) + \\ & O(m_{\text{smplx neutro}}, m_{\text{flame neutro}}) + \\ & O(m_{\text{smplx personalizado}}, m_{\text{smplx neutro}}) \end{aligned} \quad (3.3)$$

Por otro lado, en el caso de los modelos con expresión es necesario añadir un último *offset* entre las cabezas generadas por DECA con expresión y sin expresión, cuya operación se resume en la ecuación 3.4.

$$\begin{aligned} O_E = & O_{\text{NE}} + \\ & O(m_{\text{deca}}, m_{\text{deca con expresion}}) \end{aligned} \quad (3.4)$$



(a) Parámetros neutros.

(b) Parámetros deseados.

Figura 3.15: Corrección de cabeza a causa de parámetros SMPL-X.

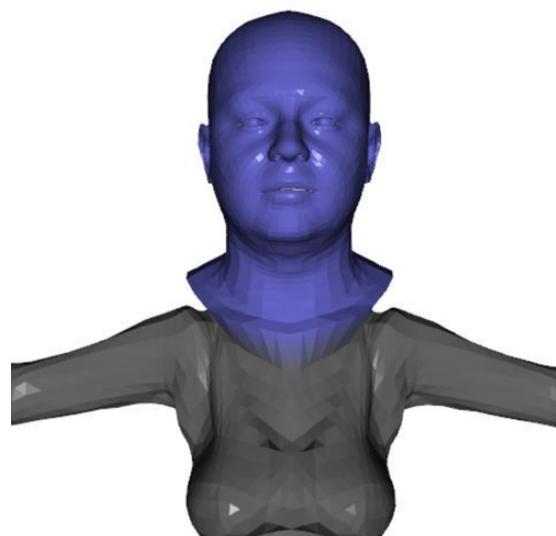


Figura 3.16: Preservación de indentidad con parámetros poco realistas.

### Suavizado de clavícula

En la Figura 3.15b se puede observar que la unión entre la cabeza y el cuerpo no se realiza de manera adecuada. Esto se debe a que la cabeza generada por DECA puede no coincidir con la cabeza generada por SMPL-X para los parámetros de cuerpo deseados, lo que provoca un desajuste en la región de unión al calcular los *offsets* y aplicarlos a la cabeza de SMPL-X.



Figura 3.17: Modelo sin suavizado de clavícula/cuello.

Para abordar este problema, se aplica un suavizado en la región del cuello mediante la asignación de pesos a los vértices de cada malla de forma lineal, lo que permite una transición gradual de un modelo a otro. En consecuencia, en la zona cercana a la clavícula se asignan pesos que corresponden principalmente a la cabeza de SMPL-X, y a medida que se acerca al cuello, se asignan pesos que se corresponden más con la cabeza de DECA. De esta manera, se logra una transición suave entre ambas cabezas y se mejora la unión entre la cabeza y el cuerpo.

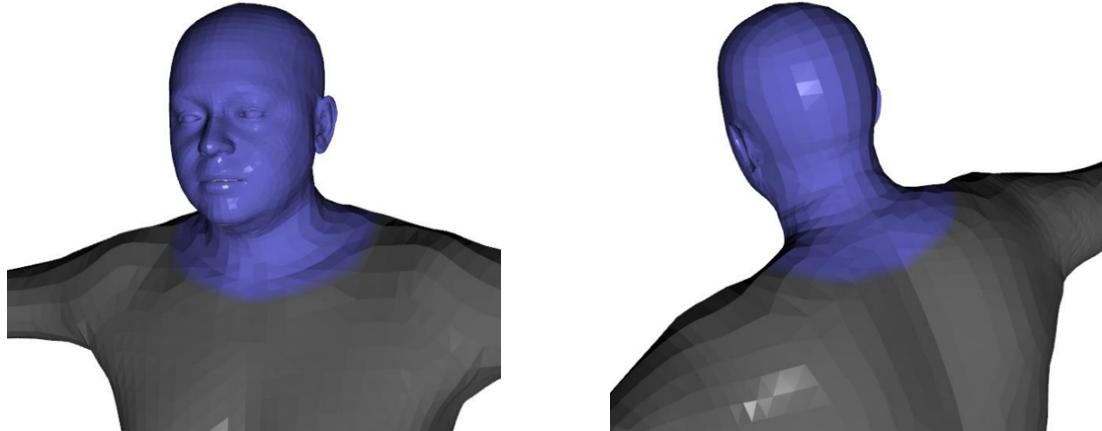


Figura 3.18: Modelo con suavizado de clavícula/cuello.

La dificultad en este proceso ha radicado en la determinación de los pesos a los vértices adecuados. La dificultad en este proceso ha sido determinar qué vértices están involucrados dentro de todo el cuerpo SMPL-X. Afortunadamente, se utilizó un *add-on* proporcionado por la empresa SEDDI, lo que facilitó considerablemente el trabajo. Este add-on permite exportar un archivo binario que contiene una lista de todos los vértices de una malla que han sido marcados en Blender con el color verde, junto con su valor asociado. Estos valores, que van de 0 a 1, representan los pesos correspondientes. En la Figura 3.19 se muestra un ejemplo del proceso de trabajo.

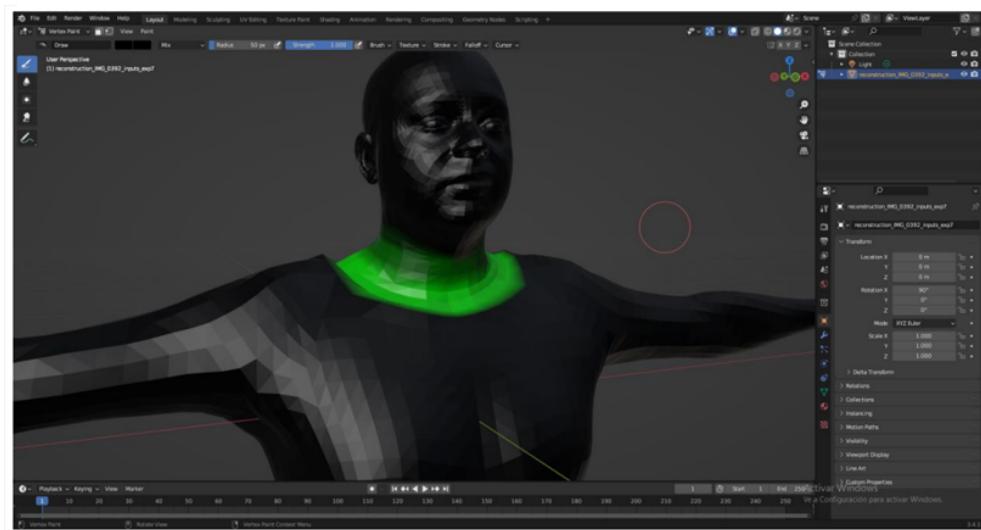


Figura 3.19: *Add-on* de Blender para la aplicación de pesos sobre los vértices.

### Inferencia de cuerpo

En la Figura 3.16 se puede apreciar que la cabeza no se ajusta adecuadamente al cuerpo deseado en términos de escala. Aunque se trata de un cuerpo no realista, este problema evidencia claramente la cuestión, que ocurre en menor medida con cuerpos realistas. Por lo tanto, al seleccionar los parámetros del cuerpo, es importante que el usuario elija un cuerpo apropiado en términos de tamaño. En caso de no desear asumir esta tarea, se propone de manera opcional un proceso de inferencia del cuerpo que buscará un cuerpo adecuado basándose en la cabeza generada por DECA. Este proceso implica generar sucesivamente cuerpos SMPL-X cuyas cabezas se asemejen más a la generada por DECA, variando los 10 parámetros de modificación de cuerpo de SMPL-X, inicializados a cero. Es por ello que el optimizador modifica estos 10 parámetros mediante un ciclo iterativo que trata de reducir la función de pérdida descrita en la Ecuación 3.5, que penaliza las distancias entre los pares de vértices.

$$Loss = \sum_{i=1}^{5023} (v_{i,\text{malla}_1} - v_{i,\text{malla}_2})^2 \quad (3.5)$$

En cada iteración, hasta un número máximo preestablecido, se genera un cuerpo SMPL-X con los parámetros actuales y se calcula la función de pérdida 3.5 mediante la cabeza generada por SMPL-X y la cabeza de DECA deseada. Después, en caso de que se haya conseguido alcanzar un nuevo mínimo, se modifican los parámetros de SMPL-X y se comienza un nuevo ciclo de optimización pudiendo generar un nuevo cuerpo SMPL-X con una cabeza más semejante a la de DECA. En caso de no conseguir reducir la función de pérdida durante un número determinado de ciclos, el proceso de optimización terminará. Además, se usa Adam como optimizador debido a su eficiencia y mayor probabilidad de convergencia en un mínimo global.

Una vez se complete la optimización, el cuerpo generado por SMPL-X presentará una cabeza de tamaño y forma similar a la de DECA, generando un cuerpo plausible, como puede observarse en la Figura 3.20, teniendo a la izquierda el cuerpo SMPL-X neutro y a la derecha el generado mediante el optimizador.

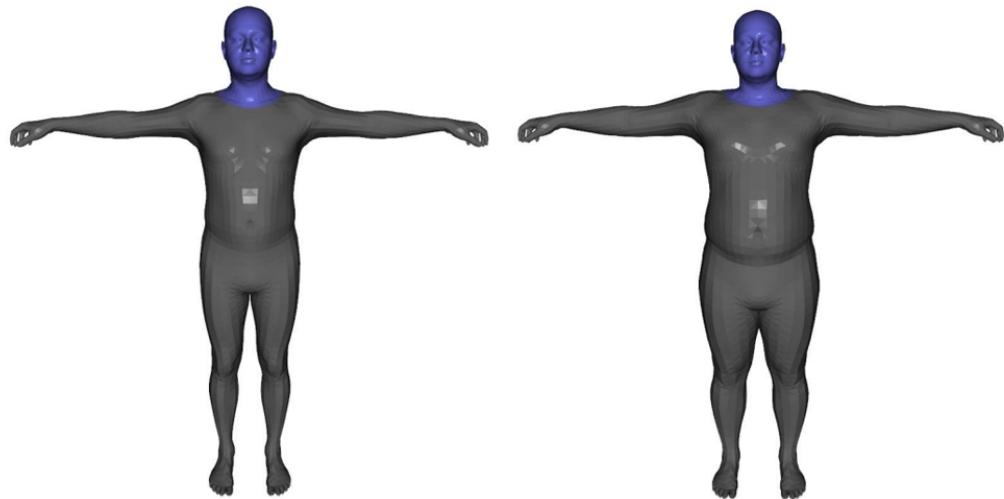


Figura 3.20: Comparación de cuerpo neutro y cuerpo generado mediante optimizador.

Para evidenciar aún más los resultados del inferenciador, en la Figura 3.21 se aplica un escalado manual de la cabeza generada por DECA. Se puede observar a la izquierda que el cuerpo nuelto no encaja bien con dicha cabeza pero, en cambio, el cuerpo optimizado es apropiado con dicha cabeza.

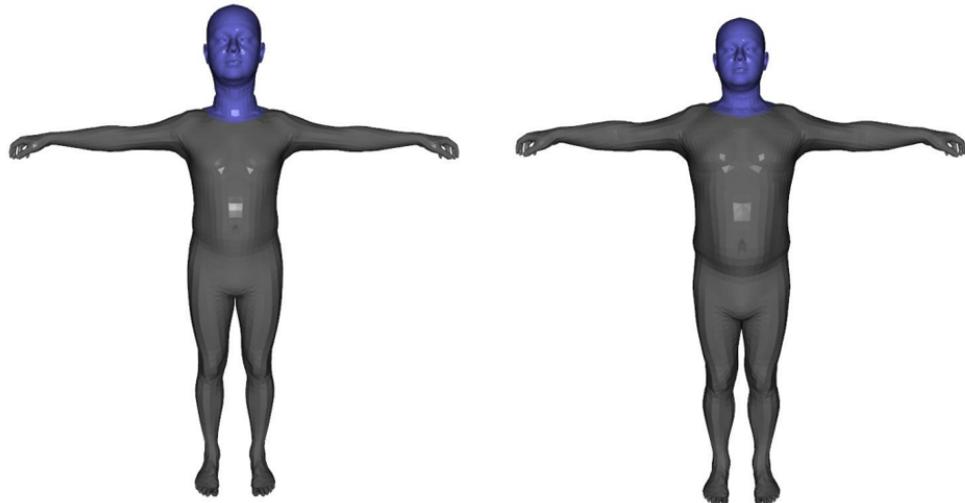


Figura 3.21: Comparación de cuerpo neutro y cuerpo generado mediante optimizador con cabeza escalada.

### Corrección de pose

A causa de que el modelo DECA permite la manipulación de la pose, la malla generada también se adaptará a la pose basada en la imagen de entrada. Este efecto no se había observado previamente debido a que la imagen utilizada anteriormente para DECA mostraba al modelo mirando directamente a la cámara. Sin embargo, si se utiliza una imagen de entrada en la cual la persona mira hacia arriba a la izquierda (su izquierda), la cabeza generada imitará esa pose, como se puede apreciar en la Figura 3.22, que también está incorporada en el modelo SMPL-X. Además, se puede observar que la boca está abierta, un efecto resultante de este mismo problema cuando la imagen de entrada que define la identidad ya tiene una cierta pose.

DECA opera internamente mediante un codificador que toma como entrada la imagen del usuario y luego decodifica estos valores para generar los datos deseados. En este proceso de codificación, existen varios valores que se asignan a propósitos específicos. Por lo tanto, para resolver este problema, simplemente es necesario establecer en cero los valores correspondientes a la pose y la expresión. Al realizar esta modificación, se obtiene el modelo mostrado en la Figura 3.23.

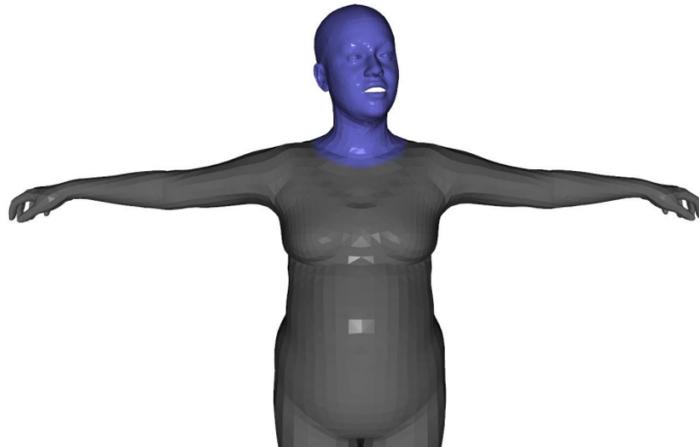


Figura 3.22: Cuerpo SMPL-X con cabeza posada.

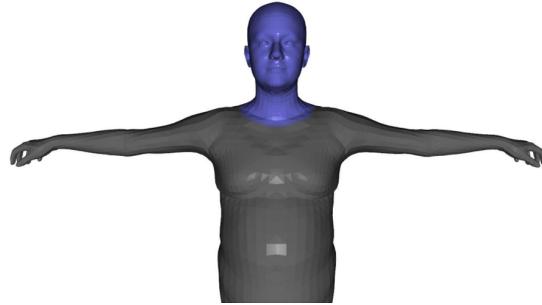


Figura 3.23: Modelo SMPL-X eliminando información de pose y de expresión residual.

Esta mejora resulta fundamental para lograr un uso adecuado de la herramienta de inferencia de cuerpo, dado que SMPL-X no puede encontrar un cuerpo realista basado en una cabeza posada. Este inconveniente se debe a que SMPL-X se instancia inicialmente en una pose neutra y la pose se aplica como un paso final en todo el proceso, lo cual puede ocasionar errores en la inferencia debido a que se basa en la correcta alineación de las cabezas. El efecto de este problema se puede apreciar en la Figura 3.24.

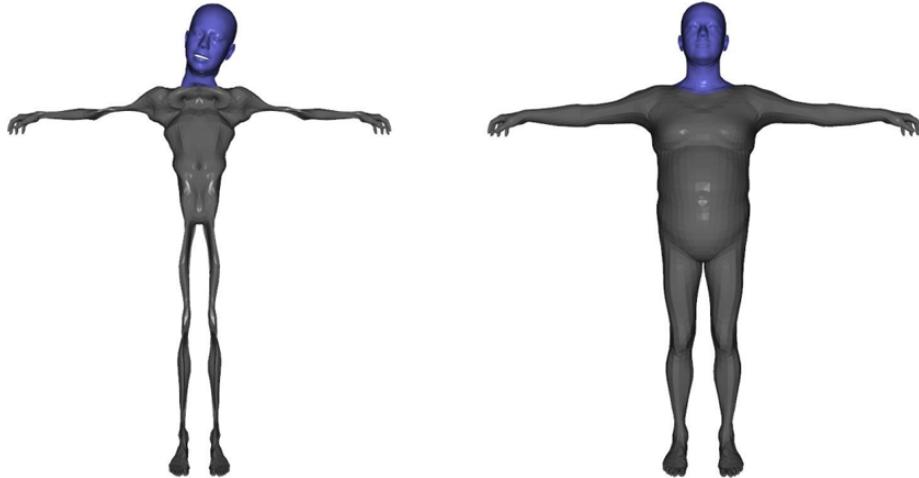


Figura 3.24: Inferencia de cuerpo con cabeza posada y cabeza no posada.

### Mejora en la alineación de cabezas mediante optimizador.

El enfoque anterior se basa en la alineación de las cabezas utilizando el centro de gravedad, es decir, el promedio de todos los vértices que conforman la malla. Inicialmente, se consideraba que esta alineación era precisa y que los modelos estaban perfectamente alineados. Sin embargo, esta alineación no minimiza la suma de las distancias entre todos los pares de vértices. Por lo tanto, en este nuevo enfoque se busca lograr la mínima distancia a través de un proceso de optimización. Se parte de la alineación obtenida en el enfoque anterior, ya que proporciona una buena aproximación inicial, y se modifica la posición de una de las cabezas de tal manera que minimice la función de pérdida de la Ecuación 3.5.

De esta manera se optimizan 3 parámetros, que son las coordenadas que representan el pivote de la malla en el espacio tridimensional, gracias al uso de la función de pérdida que, debido al uso de la exponenciación, enfatiza la importancia de los errores más grandes y los penaliza de manera proporcional. Además, se usa Adam como optimizador debido a su eficiencia y mayor probabilidad de convergencia en un mínimo global. Este proceso de optimización se realiza iterativamente hasta obtener la mejor alineación posible entre las cabezas o hasta que se ejecutan un determinado número de ciclos sin presentar una disminución en la función de pérdida.

### Interfaz para la creación de cuerpo

En caso de desear una personalización del cuerpo a medida, es decir, seleccionando manualmente los parámetros, puede resultar tedioso realizarlo sin tener conocimiento del impacto que dichas variaciones tendrán en el resultado final del cuerpo. Por lo tanto, surge la necesidad de proporcionar un entorno gráfico interactivo que permita la modificación de los parámetros del cuerpo SMPL-X, y al mismo tiempo visualizar los cambios en tiempo real. De esta manera, es posible seleccionar individualmente los 10 parámetros que definen la forma del cuerpo y ajustarlos según las necesidades específicas del usuario. Una vez se obtengan los parámetros deseados, estos podrán ser exportados para su uso posterior. Este apartado se basa en el trabajo realizado por Muhammed Kocabas, cuyo trabajo puede ser consultado en el siguiente [enlace](#)[1], el cuál ha sido modificado ligeramente con el objetivo de presentar únicamente las herramientas necesarias para este trabajo.

La interfaz gráfica de usuario (GUI, por sus siglas en inglés, Graphical User Interface) creada también permite la creación y exportación de expresiones. Aunque DECA se encarga de generar modelos con expresión, estas expresiones pueden ser modificadas gracias a estos parámetros de SMPL-X. En caso de no modificar estos parámetros, la expresión del modelo final será la generada por DECA sin ninguna alteración.

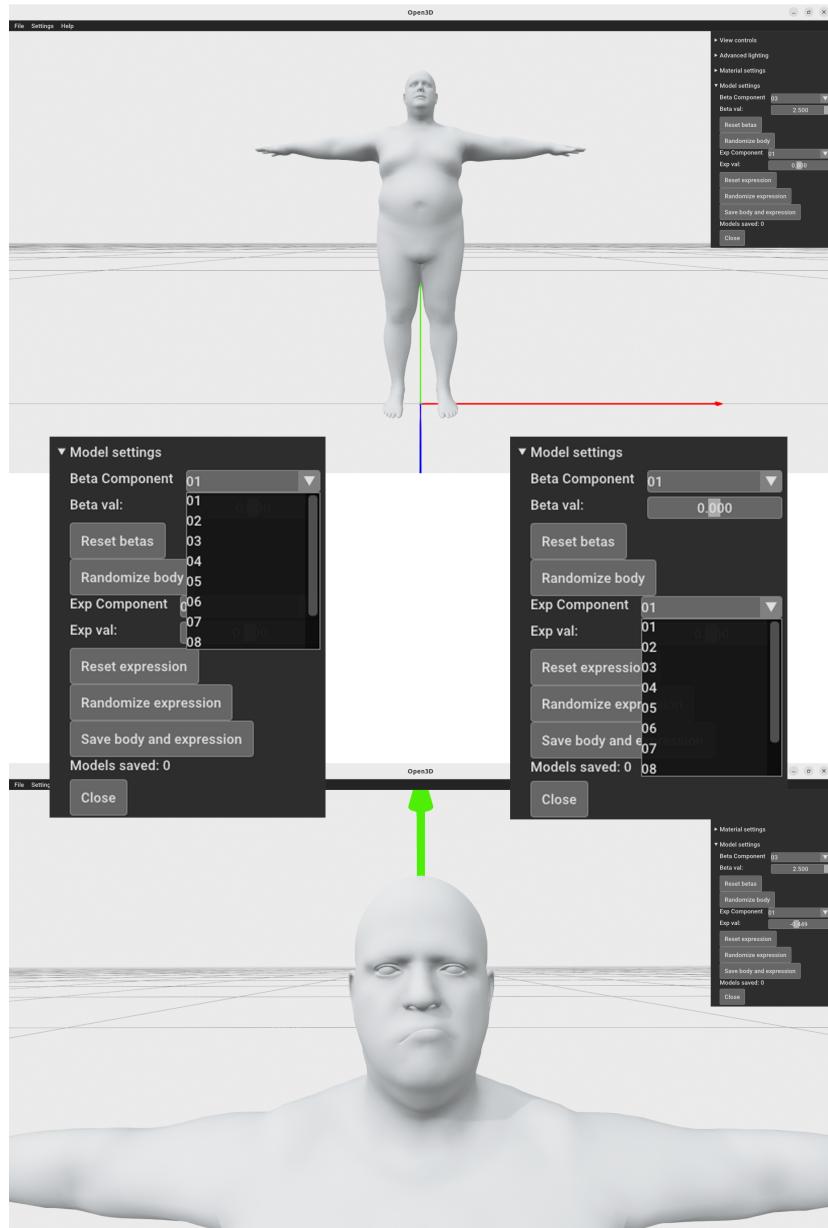


Figura 3.25: Creación de cuerpo y expresión SMPL-X mediante interfaz gráfica de usuario.

### Pipeline de geometría

El flujo de trabajo del proceso concerniente a la geometría se puede ver resumido en la Figura 3.26.

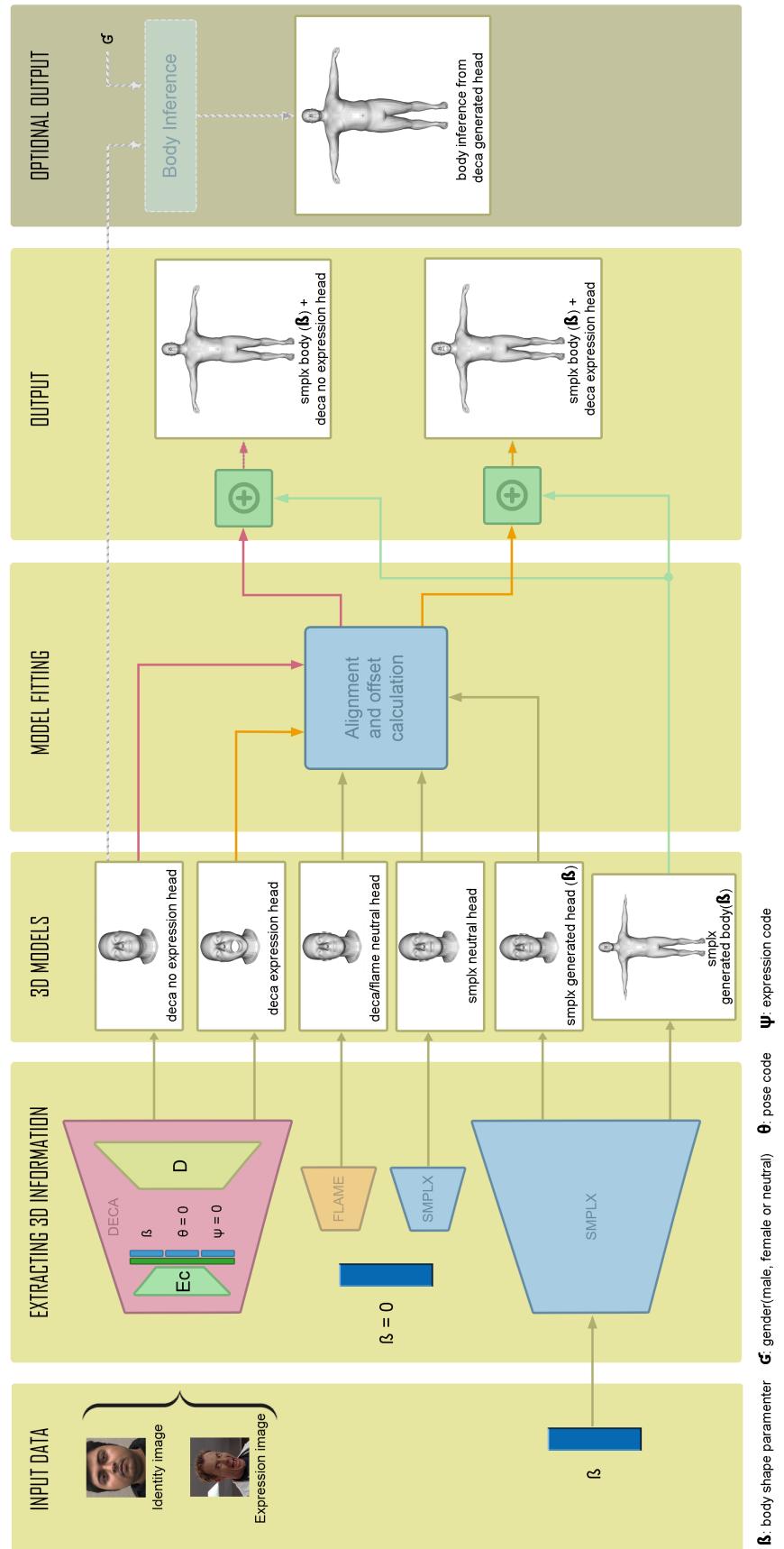


Figura 3.26: Pipeline de geometría.

### 3.3. Tratamiento de texturas

DECA, además de generar una geometría, posee la capacidad de crear múltiples texturas basadas en la imagen de entrada mediante un proceso de renderizado inverso. A pesar de su alto rendimiento, se han identificado ciertas limitaciones y áreas de mejora en estas texturas, las cuales se discuten en este apartado con detalle.



Figura 3.27: Comparación de cabeza DECA con y sin mapa de albedo. Vista frontal.

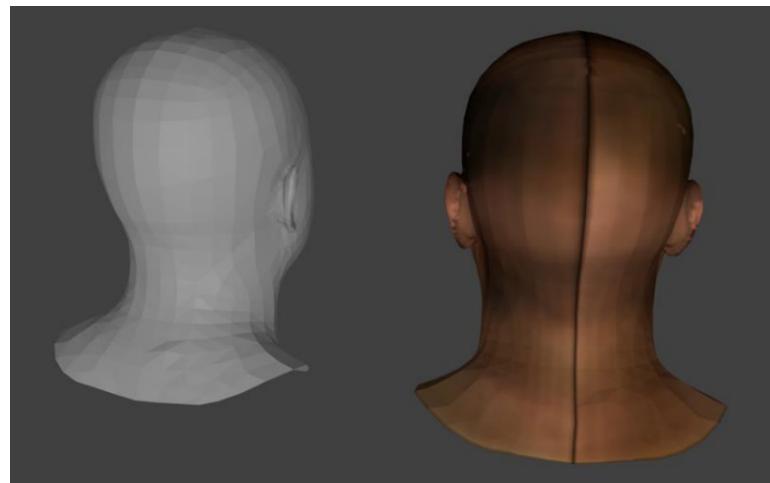


Figura 3.28: Comparación de cabeza DECA con y sin mapa de albedo. Vista trasera.

Por otro lado, DECA genera un mapa de normales mediante una malla de alta resolución de 55.000 vértices, que crea de forma interna, lo que permite capturar con mayor detalle las imperfecciones y arrugas de la piel. En las Figuras 3.29 y 3.30 se presentan dos mallas con la misma topología, siendo la de la derecha la que ha sido aplicada con el mapa de normales.

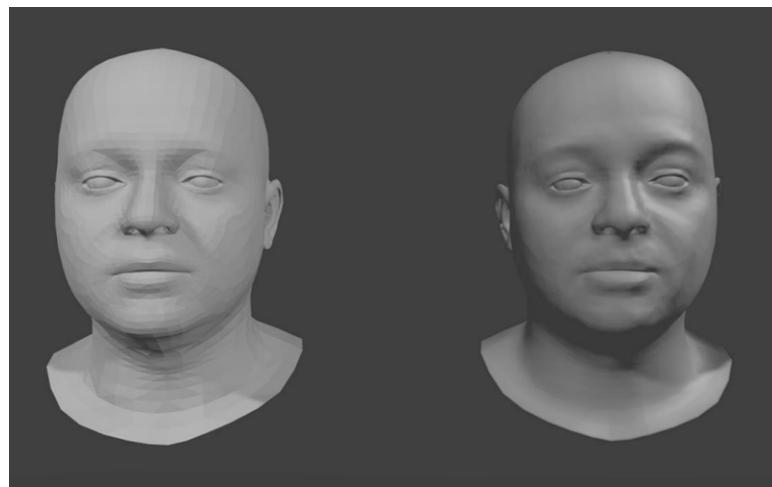


Figura 3.29: Comparación de cabeza DECA con y sin mapa de normales. Modelo con expresión neutra.

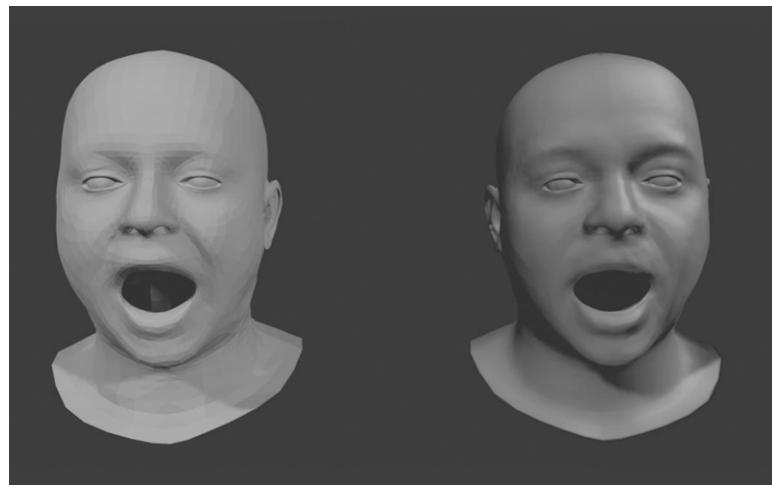


Figura 3.30: Comparación de cabeza DECA con y sin mapa de normales. Modelo con expresión personalizada.

Por consiguiente, en la Figura 3.31 se pueden observar las diferencias al aplicar el mapa de normales al modelo junto con el albedo. También se presenta una problemática, la cual se abordará posteriormente, en la zona de los ojos. En cada ejecución, DECA realiza una predicción de *landmarks*, siendo estos los puntos más significativos en la imagen, que definen diversas áreas de la cara y son utilizados posteriormente para generar las texturas correspondientes. En algunos casos, como en este ejemplo, se produce un desfase en la zona de los ojos, lo que ocasiona que la textura no se ajuste correctamente.



Figura 3.31: Comparación de cabeza DECA con y sin mapa de normales y con mapa de albedo.

### Corrección en la proyección de la textura

Al eliminar anteriormente la información de pose y expresión, DECA ya no es capaz de proyectar correctamente la textura sobre el modelo.



Figura 3.32: Textura mal proyectada.

Durante el proceso de codificación en DECA, se genera un espacio latente de 236 valores que contienen información importante para la decodificación. Tres de estos valores corresponden a propiedades específicas de la cámara, como el factor de escalado isotrópico y la traslación 2D, los cuales se utilizan en el proceso de rendering inverso. Por lo tanto, este problema no se resuelve simplemente estableciendo a cero dichos valores, como se hizo con los valores de pose y expresión.

Para abordar este problema, se ha optado por realizar el proceso de decodificación dos veces. En la primera decodificación, los parámetros de pose y expresión no se establecen a cero, lo que resulta en texturas generadas correctamente por DECA, pero se obtiene una malla indeseada. En la segunda decodificación, los parámetros de pose y expresión se establecen a cero, lo que produce la malla deseada pero texturas incorrectas. Al combinar los componentes adecuados de cada decodificación, se resuelve el problema, como se puede apreciar en la Figura 3.33.



Figura 3.33: Textura bien proyectada.

### Texturizado de cabeza en SMPL-X

Debido a que DECA es capaz de generar una textura y aplicarla a su modelo, el siguiente paso lógico es conseguir este texturizado de cabeza sobre el cuerpo SMPL-X. Para lograr esto, es necesario realizar un remapeo de las coordenadas UV correspondientes a la parte de la cabeza de la textura SMPL-X hacia la textura generada por DECA. Para abordar este problema, se ha utilizado el enfoque implementado en el proyecto [FLame2SMPLX\[9\]](#). Este enfoque permite generar la textura mostrada en la Figura 3.35 a partir de la textura generada por DECA, que se muestra en la Figura 3.34, obteniendo los resultados presentados en la Figura 3.36.



Figura 3.34: Textura de albedo generada por DECA.

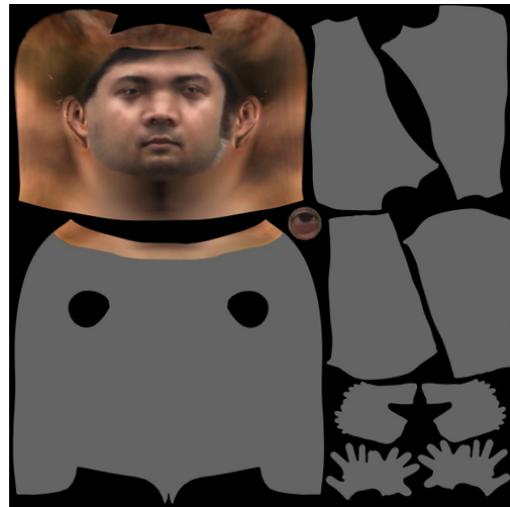


Figura 3.35: Textura de albedo generada por [FLame2SMPLX](#).



Figura 3.36: Texturizado SMPL-X mediante [FLame2SMPLX](#).

Con el enfoque actualmente utilizado, surge un problema significativo relacionado con el tiempo de ejecución, lo que dificulta su integración en cualquier flujo de trabajo. El proceso de generación de cada textura requiere un tiempo promedio de 13 minutos. Esta demora se debe al algoritmo implementado, que realiza un warping de todas las caras topológicamente hablando, de una textura a otra. Considerando que además de la textura de albedo también se debe generar el mapa de normales, el tiempo de ejecución se incrementa aún más. Con el objetivo de reducir el tiempo de ejecución, se realizaron modificaciones en ciertas partes del código y se implementó el paralelismo utilizando la biblioteca joblib[6] de Python. Gracias a estas optimizaciones, ahora es posible generar las texturas de albedo y normales en 3 minutos. Aunque se ha logrado una mejora significativa en el tiempo de ejecución, aún no es suficiente para su integración en un entorno de producción real. Por ello, se busca otra solución.

Gracias a las correspondencias entre los vértices de las cabezas de ambos modelos, es posible determinar qué coordenada UV de un modelo corresponde a otra en el otro modelo. Basándonos en esta premisa, se ha adoptado un nuevo enfoque en el cual se combinan las texturas de las Figuras 3.37 y 3.34 en una única textura. Además, se ajustan las coordenadas UV correspondientes a la cabeza del modelo SMPL-X para que se alineen correctamente con la textura generada por DECA. El resultado de este proceso se muestra en las Figuras 3.38 y 3.39, donde se aprecian las texturas resultantes.

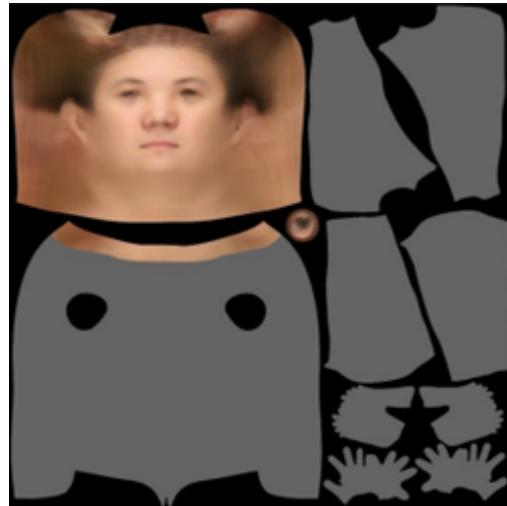


Figura 3.37: Textura de albedo genérica de SMPL-X utilizada como plantilla.



Figura 3.38: Combinación de texturas de albedo SMPL-X y DECA.

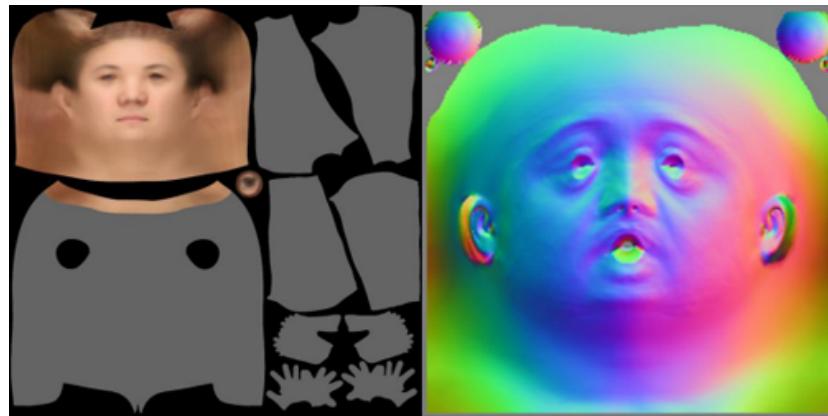


Figura 3.39: Combinación de texturas de mapa de normales de SMPL-X y DECA.

Gracias a este nuevo algoritmo el programa es capaz de generar ambas texturas en apenas un segundo.

### ***Inpainting***

Aunque actualmente existen una gran variedad de algoritmos de *inpainting*, en este apartado se implementa un método básico y de bajo costo computacional, basado en la vecindad de los píxeles cercanos. Por su simplicidad, resulta más bien, un ejercicio académico sobre procesamiento de imagen a nivel de píxel, siendo más efectivo el uso de otros métodos, basados en librerías existentes, que se mencionan al final de esta sección.

En función de la imagen de entrada utilizada, DECA genera texturas de albedo, como se muestra en la Figura 3.34. Sin embargo, en algunos casos particulares, debido a una predicción incorrecta de *landmarks*, se produce un artefacto negro en la región de la frente, como se observa en la Figura 3.40. Esto resulta en la generación del modelo SMPL-X que se muestra en la Figura 3.41.

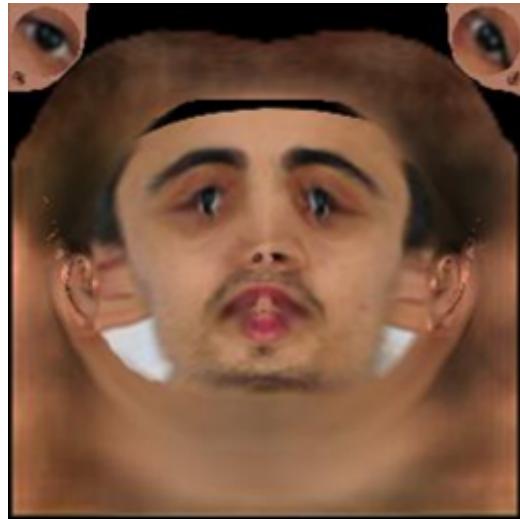


Figura 3.40: Textura DECA con zona oscura en la zona de la frente.



Figura 3.41: Modelo SMPL-X texturizado con zona oscura en la zona de la frente.

Con el objetivo de abordar este problema, se ha desarrollado una herramienta opcional que se encarga de solucionar el artefacto en la frente mediante el relleno de píxeles. Para utilizar esta herramienta, el usuario debe seleccionar cuidadosamente un área específica que cubra la zona negra de la frente, como se muestra en la Figura 3.42. La selección manual de esta área se debe a que su ubicación varía en función de la textura generada por DECA para cada sujeto en particular.

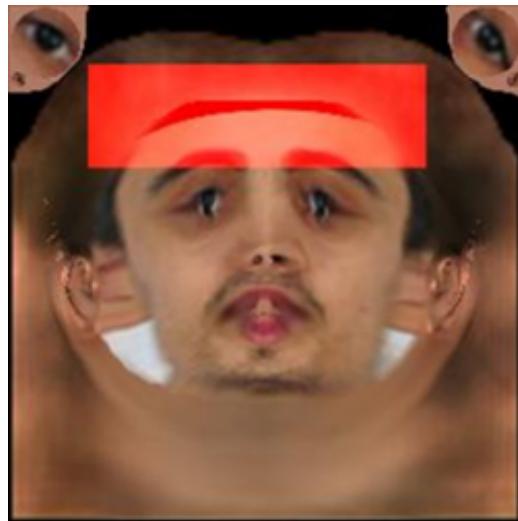


Figura 3.42: Selección manual de área oscura.

A continuación, se utiliza la fórmula

$$factor = \sum_{x=1}^n \sum_{y=1}^m (0,2126 \cdot B_{x,y} + 0,7152 \cdot G_{x,y} + 0,0722 \cdot R_{x,y}) \quad (3.6)$$

dónde RGB son los valores de los canales de color, para determinar el factor de luminancia de la región. Este factor juega un papel crucial en la identificación de los píxeles pertenecientes a la zona oscura dentro de dicha región. Utilizando esta información, se procede a procesar todos los píxeles cuyos valores sean inferiores al factor de luminancia. En el caso de encontrar uno de estos píxeles, se realiza un cálculo para determinar un nuevo color basado en la combinación ponderada de dos píxeles adyacentes, que se encuentren tanto por encima como por debajo del píxel en consideración, y que deben cumplir la condición de estar por encima del factor de luminancia. La ponderación se realiza teniendo en cuenta la distancia entre los píxeles hallados, proporcionando el color final.

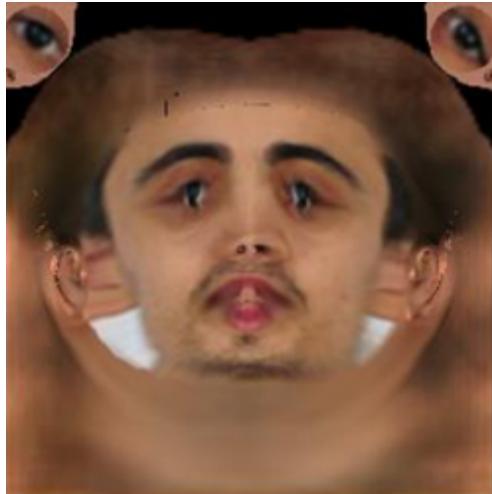


Figura 3.43: Textura de DECA con relleno de píxeles en la zona de la frente.



Figura 3.44: SMPL-X con relleno de píxeles en la zona de la frente.

Como métodos de mejora se podrá implementar el algoritmo descrito [aquí](#)[5] o el uso de la herramienta de *inpaintng*[4] de la librería de *OpenCV*[8].

### Rellenado y suavizado de bordes

En las texturas mostradas en las Figuras 3.34 y 3.40, se puede observar la presencia de bandas negras en los bordes laterales e inferior, lo que impide que la textura alcance los límites al completo. Este inconveniente es problemático ya que las coordenadas UV se extienden hasta los bordes de la textura, y es consistente en todos los modelos generados, lo que resulta en un efecto de costura en la región de la coronilla y la parte trasera, como se muestra en la Figura 3.45.

En consecuencia, se procede a realizar un proceso de interpolación para llenar los píxeles negros de la imagen. Este proceso implica realizar una ponderación entre los dos píxeles coloreados más cercanos, lo cual contribuye a generar la textura que se muestra en la Figura 3.46.

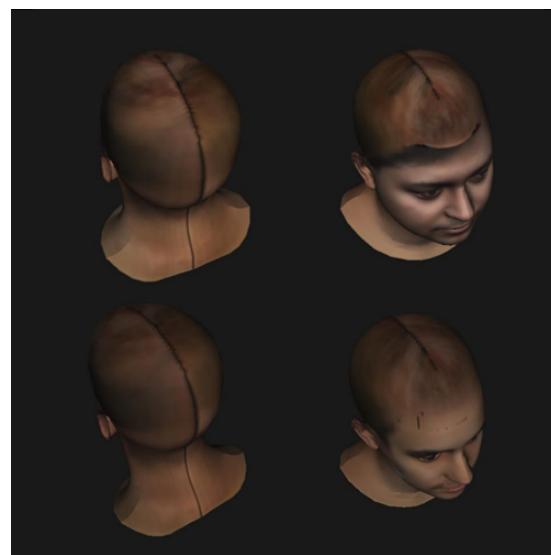


Figura 3.45: Efecto de costura en texturas generadas por DECA.

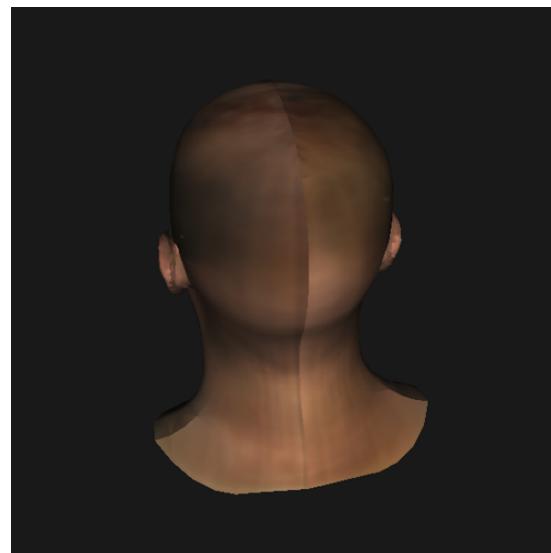


Figura 3.46: Modelo DECA con textura sin efecto costura.

Como se puede apreciar en la imagen previa, se puede notar claramente la transición de color entre diferentes zonas. Con el fin de evitar este efecto no deseado, se realiza un proceso de suavizado en la región correspondiente. Este suavizado implica la selección de pares de vértices simétricos, es decir, aquellos que se encuentran a la misma distancia del centro de la textura, y se modifica sus colores calculando el promedio de ambos. El resultado de este proceso se muestra en la Figura 3.47.

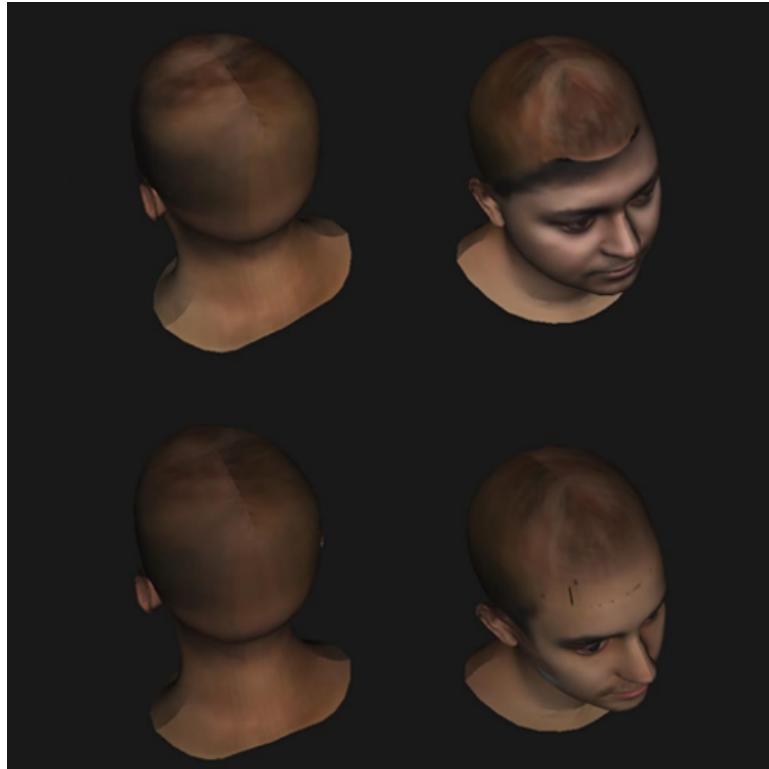


Figura 3.47: Modelos DECA con texturas suavizadas en la zona trasera.

### Corrección de textura en la parte de la clavícula

Como se puede apreciar en la Figura 3.41, se observa un efecto inusual en la zona de la clavícula de la textura. Este efecto se debe al proceso llevado a cabo en la sección 3.3, donde se combinan las texturas de DECA y SMPL-X en una sola y se realizan modificaciones en las coordenadas UV correspondientes a la cabeza de SMPL-X.

Para solucionarlo, se localizan los pares de vértices correspondientes y se juntan en una posición intermedia. En la Figura 3.48 se muestra el proceso, en el que se localizan los vértices, las correspondencias y se unen finalmente.



Figura 3.48: Proceso de unión de vértices en la zona de la clavícula para corrección de textura.

### Alineación de ojos

Uno de los desafíos principales de DECA, que varía en intensidad dependiendo de la imagen de entrada, es lograr una alineación adecuada de la textura en la región de los ojos. Aunque en la Figura 3.44 no se aprecia esta desalineación, en la Figura 3.33 se puede observar claramente que los ojos presentan un desfase entre la topología y la textura. Inicialmente se creyó que este desfase se debía a una mala predicción de los *landmarks*, por lo que se modificó el proceso de entrenamiento de DECA con el objetivo de hacer un *overfitting* del modelo en base a la función de error definida por dichos *landmarks* con el objetivo de generar una textura perfecta. Para ello, el objetivo planteado fue modificar los píxeles de la foto de entrada de tal manera que minimizasen la función de error de los *landmarks*, definida en el *paper* original de DECA[19]. Sin embargo, esta estrategia no ha mejorado la situación, lo que llevó a un análisis más detallado de cómo se produce esta textura.

Cuando DECA realiza la decodificación del espacio latente y genera toda la información requerida, la textura final no proviene únicamente de esta decodificación, sino que el decodificador genera una textura que luego es modificada por un renderizador inverso. La textura generada por el decodificador se ajusta perfectamente a la malla, pero no preserva la identidad del sujeto, sino otras características como el color de piel final, como se ilustra en la Figura 3.49.

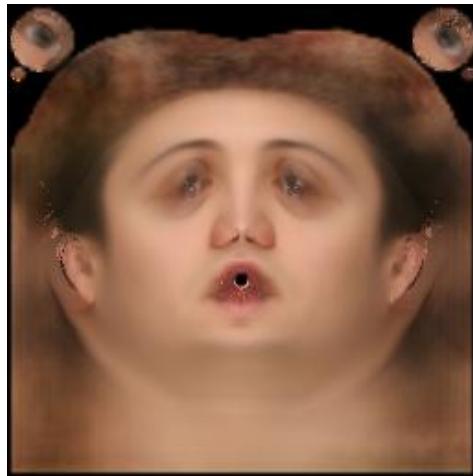


Figura 3.49: Imagen generada por el decodificador de DECA.

Las características mencionadas se combinarán con otras texturas generadas por el renderizador, utilizando una máscara, lo que provoca el desfase observado. Para una mejor comprensión, se presenta de manera resumida el proceso seguido.

Una vez obtenida la textura de la Figura 3.49, el renderizador realiza el sombreado, generando la textura de la Figura 3.50, que posteriormente se combina con la textura de la Figura 3.49, resultando en la textura de la Figura 3.51.



Figura 3.50: Imagen generada por el renderizador de DECA que guarda la información del sombreado.



Figura 3.51: Albedo generado mediante textura decodificada y sombreado de renderizador.

Por otro lado, el renderizador produce una textura *ground truth*, mostrada en la Figura 3.52, basada en la imagen de entrada de DECA, es decir, la proporcionada por el usuario. Esta textura generada presenta el desfase en los ojos que posteriormente se reflejará en el modelo. Finalmente, las texturas de las Figuras 3.52 y 3.51 se combinan utilizando una máscara (Figura 3.53), proporcionada por DECA, resultando en la textura final que se muestra en la Figura 3.54 y se utiliza en el modelo.



Figura 3.52: Textura *ground truth* generada por el renderizador de DECA.



Figura 3.53: Máscara predeterminada de DECA.



Figura 3.54: Textura de albedo generada por DECA para utilizar sobre el modelo.

Por lo tanto, al realizar un ajuste en la máscara utilizada, es posible preservar correctamente la zona de los ojos de la Figura 3.51, la cual se sabe que se ajusta bien a la malla. De esta manera, se crea manualmente la máscara representada en la Figura 3.55, la cual se utilizará para generar la textura final mostrada en la Figura 3.56.



Figura 3.55: Máscara personalizada para corrección de ojos.



Figura 3.56: Textura de albedo generada por DECA utilizando la máscara personalizada.

Con respecto al modelo final, esta nueva textura soluciona el problema al garantizar una textura adecuada en la zona problemática, sin afectar la identidad de la persona, como se puede observar en la Figura 3.57. Sin embargo, surge otro inconveniente a resolver, que es la alineación de la textura en el globo ocular, el cual es un efecto secundario del problema anteriormente mencionado.



Figura 3.57: Modelo DECA usando como albedo la textura generada con la máscara personalizada.

Para ilustrar aún más este desalineamiento, en las Figuras 3.58 se presenta a la izquierda un ojo correctamente alineado, correspondiente al modelo de la Figura 3.44, y a la derecha el ojo correspondiente al modelo actual. En ambas figuras, el punto verde representa el centro ideal para cualquier ojo generado, mientras que el punto azul representa el centro deseado.

Por lo tanto, es necesario encontrar una forma de determinar el centro deseado en cada ojo y desplazar la textura para que se alinee con el centro ideal, como se muestra en las Figuras 3.59.



Figura 3.58: Ojos de la textura de albedo generada por DECA para distintos sujetos. Píxel verde: centro ideal. Píxel azul: centro deseado.

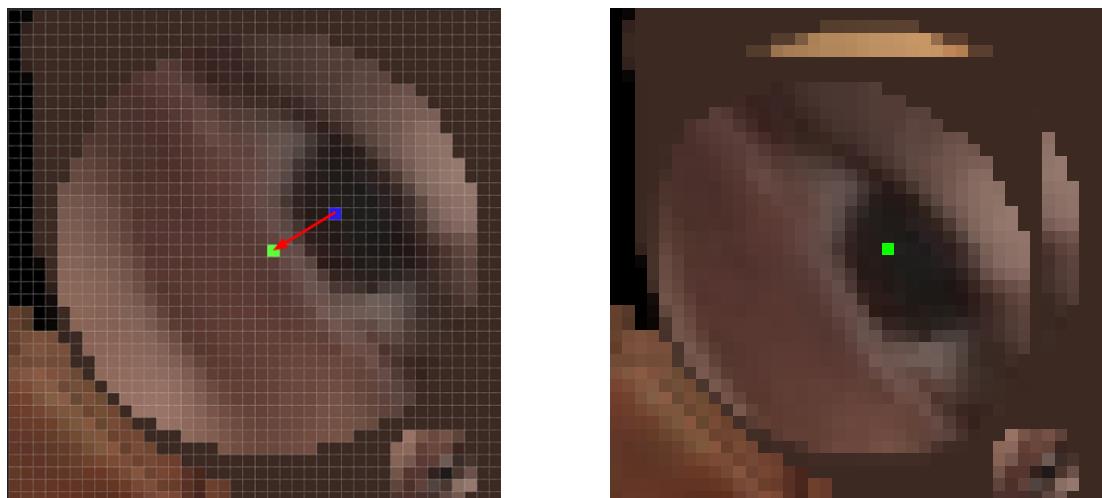


Figura 3.59: Desplazamiento de textura para corrección de ojos.

Aunque la textura desplazada genera zonas incorrectas, estas se verán tapadas por la propia geometría del modelo, por lo cual se pueden ignorar.

Con el fin de automatizar estos desplazamientos, se requiere predecir el centro deseado. Para lograr esto, se ha desarrollado una red neuronal capaz de predecir las coordenadas de los centros deseados, utilizando un conjunto de datos personalizado. Este conjunto de datos está formado por 82 recortes de ojos de las texturas de albedo generadas por DECA a partir de una gran variedad de sujetos, como se muestra en la Figura 3.60, donde los centros deseados han sido etiquetados manualmente, y las imágenes tienen el tamaño de 40x40 píxeles.

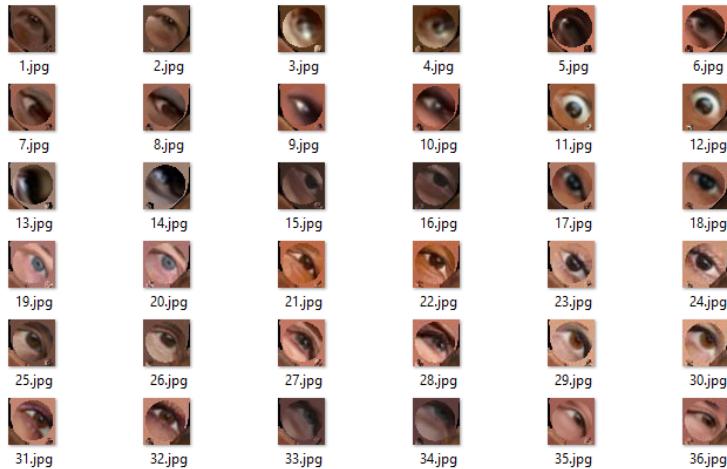


Figura 3.60: *Dataset* utilizada para el entrenamiento de red neuronal.

El modelo de red neuronal desarrollado combina dos arquitecturas principales para realizar dos tareas específicas. La primera parte de la arquitectura se compone de capas convolucionales, que están diseñadas para detectar y extraer características relevantes de las formas presentes en las imágenes de entrada. Estas capas convolucionales son muy efectivas para capturar patrones y estructuras. Una vez que se han extraído las características, la segunda parte del modelo utiliza una red neuronal de tipo MLP (*Multilayer Perceptron*) para la predicción de los píxeles del centro deseado.

La parte basada en capas convolucionales reduce progresivamente los tamaños de la imagen mediante el uso de capas *Max Pooling* que guardan las partes más representativas de los patrones hallados, a la vez que se amplian los canales de la imagen mediante uso de filtros(*kernels*) que son aprendidos en el proceso de entrenamiento. Por lo tanto, considerando que cada bloque de esta arquitectura está formada por una convolución 2d, una función de activación ReLU y una capa *Max Pooling*, los tamaños de salida de cada bloque se ven representados en la Figura 3.61, siendo el primer valor el número de canales y el segundo y tercero el tamaño en píxeles de la imagen:

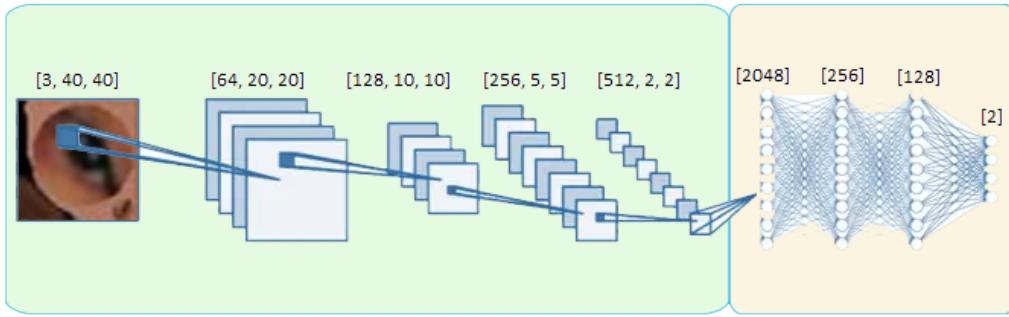


Figura 3.61: *Arquitectura de red neuronal seleccionada para el alineamiento de ojos. En verde la arquitectura basada en capas convolucionales. En naranja la arquitectura MLP completamente conectada.*

Para la segunda parte de la arquitectura, MLP completamente conectada, lo primero que se hace es aplanar el vector de salida de las capas convolucionales a un vector unidimensional que pueda ser introducido en la primera capa de MLP, obteniendo una primera capa de dimensión  $512 * 2 * 2 = 2048$ . Al igual que con las capas convolucionales, en esta parte también se reduce poco a poco la dimensionalidad del espacio mediante el uso de varias capas con dimensiones  $[2048, 256, 128, 2]$ , que finalmente predicen las coordenadas del centro deseado, siendo el primer valor la coodenada x, y el segundo valor la coordenada y. Además, a diferencia de la primera parte de la arquitectura, la MLP hace uso de capas *dropout* las cuales ayudan en el proceso de aprendizaje ya que su capacidad de generalización aumenta.

Por otro lado, la red neuronal ha sido entrenada utilizando un optimizador Adam, una función de error *MSELoss* y un *learning rate* de 0.0001 inicial. Además, el *dataset* utilizado ha sido dividido en las tres categorías clásicas de entrenamiento/validación/testing con unas cantidades proporcionales de 80/10/10 sobre el total.

Como resultado, cuando se ejecuta esta red neuronal, predice como centro el píxel azul de la Figura 3.58 y luego se realiza un desplazamiento como se muestra en la Figura 3.59. Este enfoque ha demostrado mejoras en la mayoría de los casos, como se aprecia en la Figura 3.62 para el caso en el que se ha estado trabajando a lo largo del proyecto, donde se presenta, de izquierda a derecha, la imagen de entrada, el modelo generado mediante el uso de la nueva máscara y el modelo después de aplicar la corrección en los ojos mediante la red neuronal.



Figura 3.62: Resultado del uso de corrección de ojos mediante red neuronal.

# 4

## Resultados

### 4.1. Ejecución del modelo

A continuación, se presentan diversos resultados del programa para diferentes conjuntos de datos de entrada. Estos ejemplos están organizados en dos imágenes, cada una destacando aspectos específicos de cada temática. La primera imagen se enfoca en demostrar los efectos del texturizado final en un modelo con y sin expresión que, siguiendo un orden de izquierda a derecha y de arriba hacia abajo, se muestra:

- Entrada de datos para la identidad del sujeto.
- Entrada de datos para la expresión del sujeto.
- Modelo sin expresión desde una perspectiva frontal.
- Modelo con expresión desde una perspectiva frontal.
- Modelo sin expresión desde una perspectiva trasera.

Por otro lado, la segunda imagen se centra en mostrar los resultados de la calidad topológica y, de arriba hacia abajo, se muestra:

- Cuerpo seleccionado manualmente mediante el uso de la interfaz de usuario.
- Inferencia de cuerpo a partir de cabeza generada por DECA.

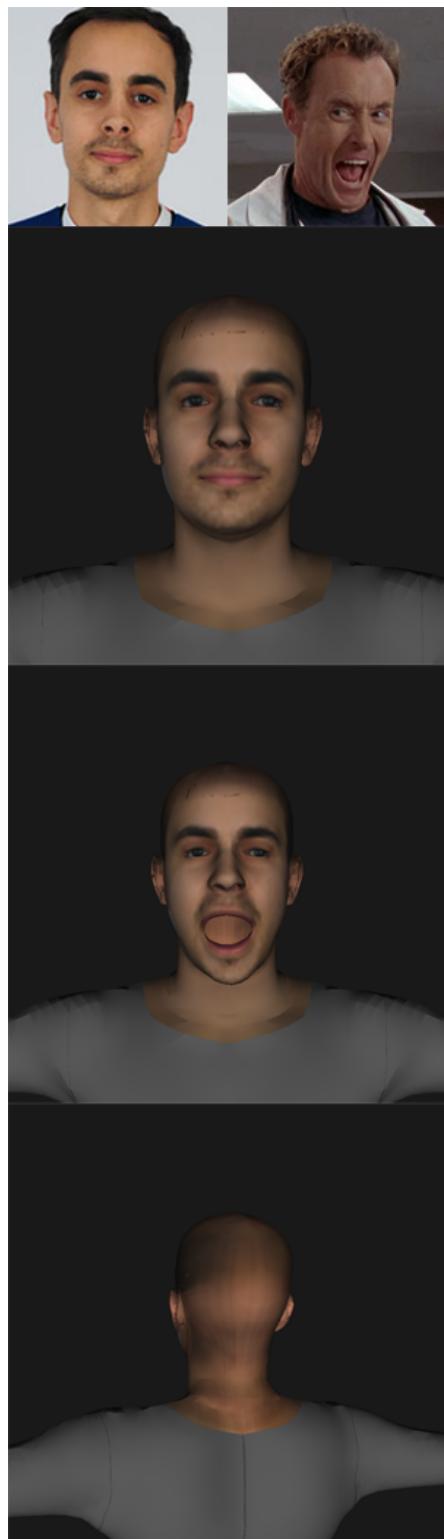


Figura 4.1: Resultados de texturizado y expresión en sujeto número 1.

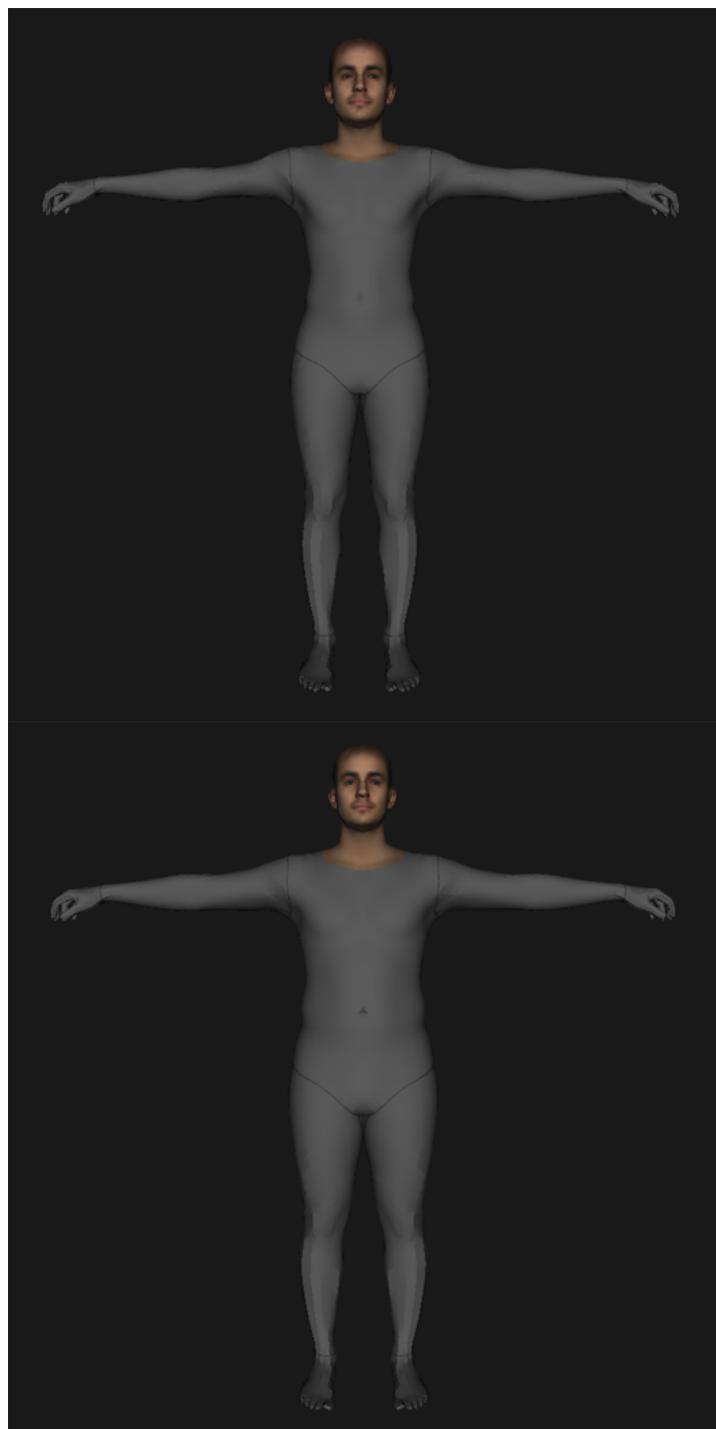


Figura 4.2: Resultados de topología en sujeto número 1.

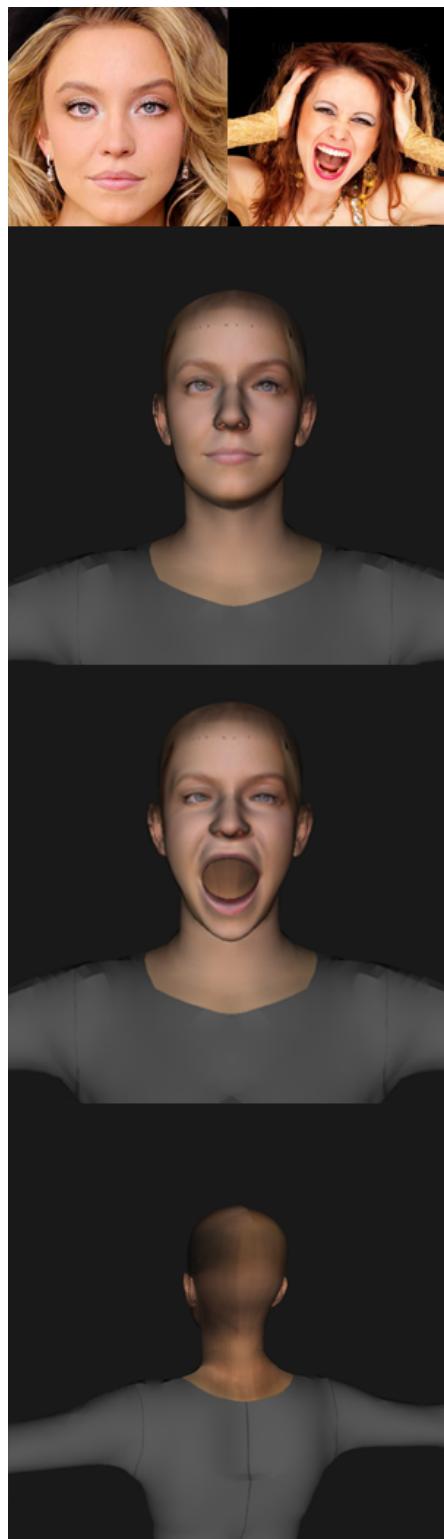


Figura 4.3: Resultados de texturizado y expresión en sujeto número 2.

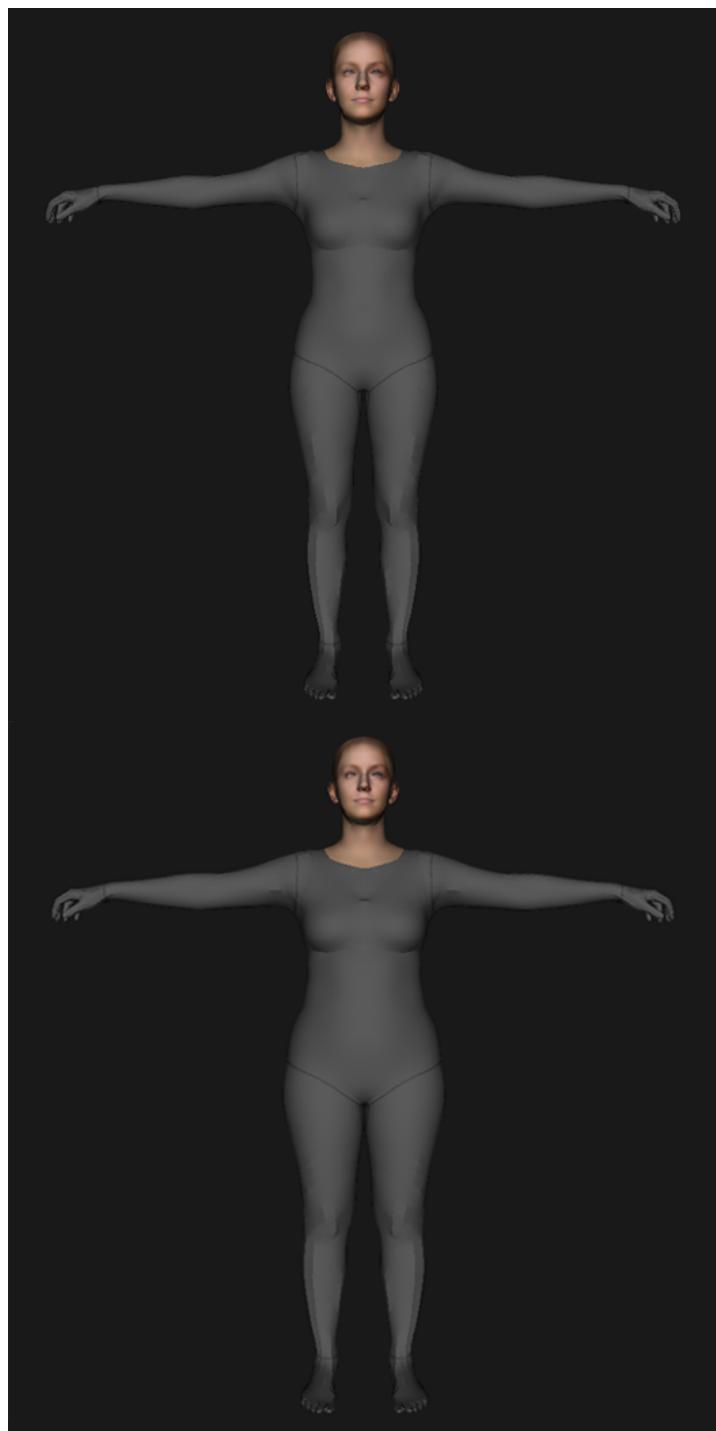


Figura 4.4: Resultados de topología en sujeto número 2.

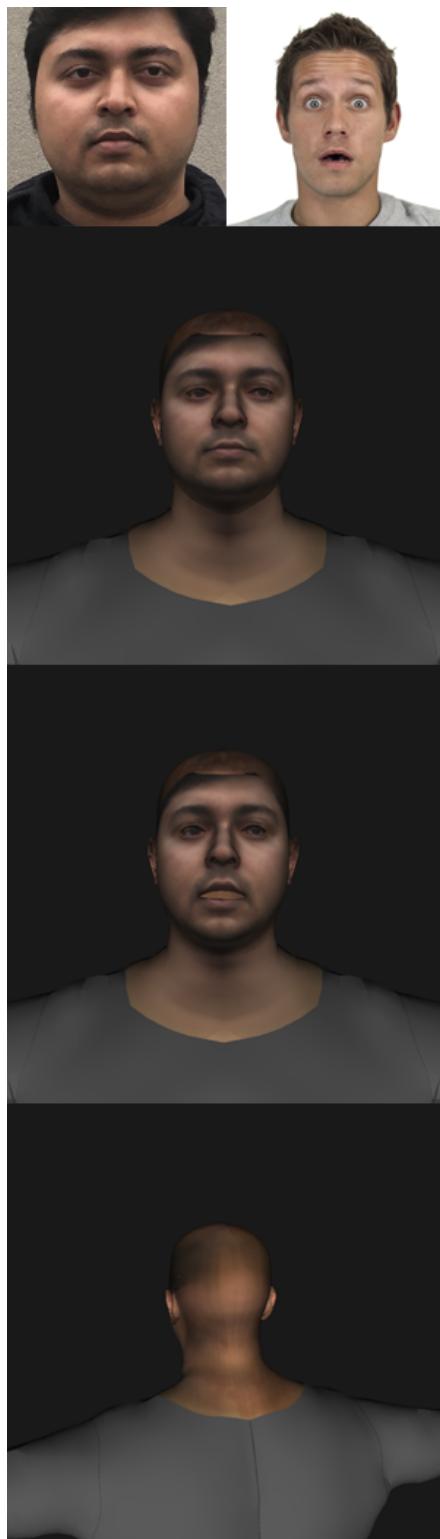


Figura 4.5: Resultados de texturizado y expresión en sujeto número 3.

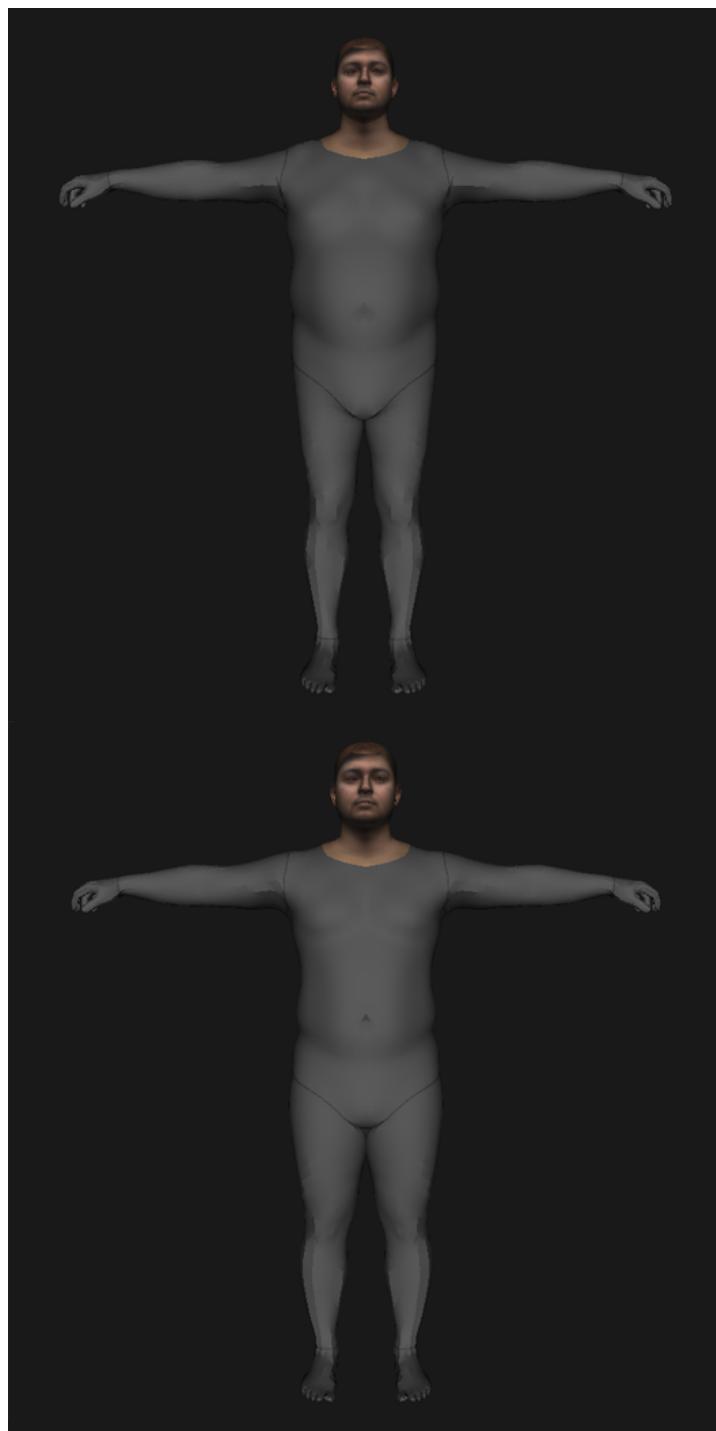


Figura 4.6: Resultados de topología en sujeto número 3.

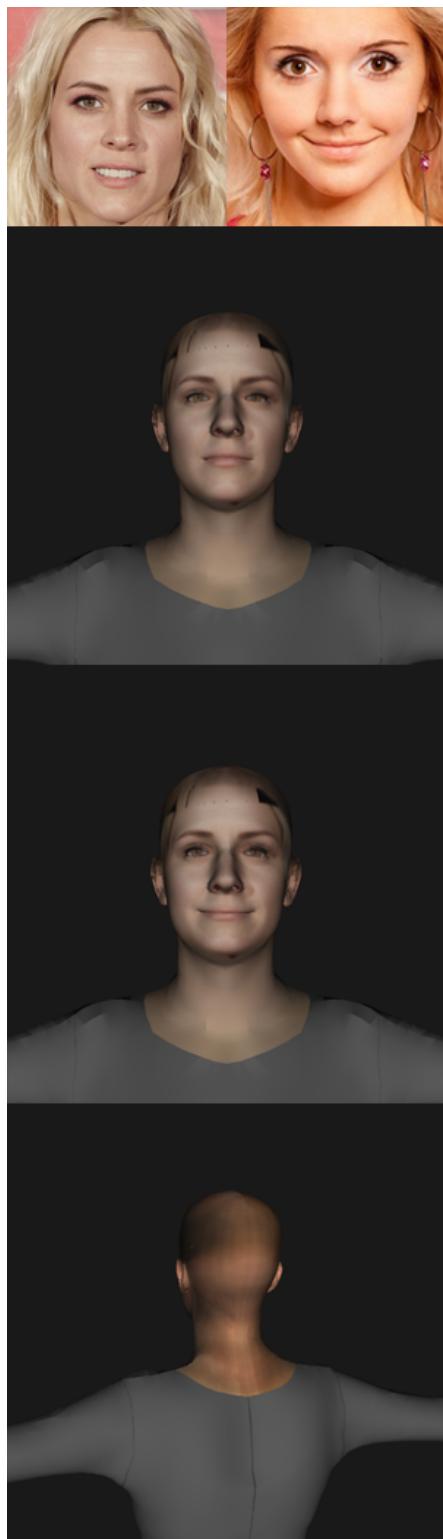


Figura 4.7: Resultados de texturizado y expresión en sujeto número 4.

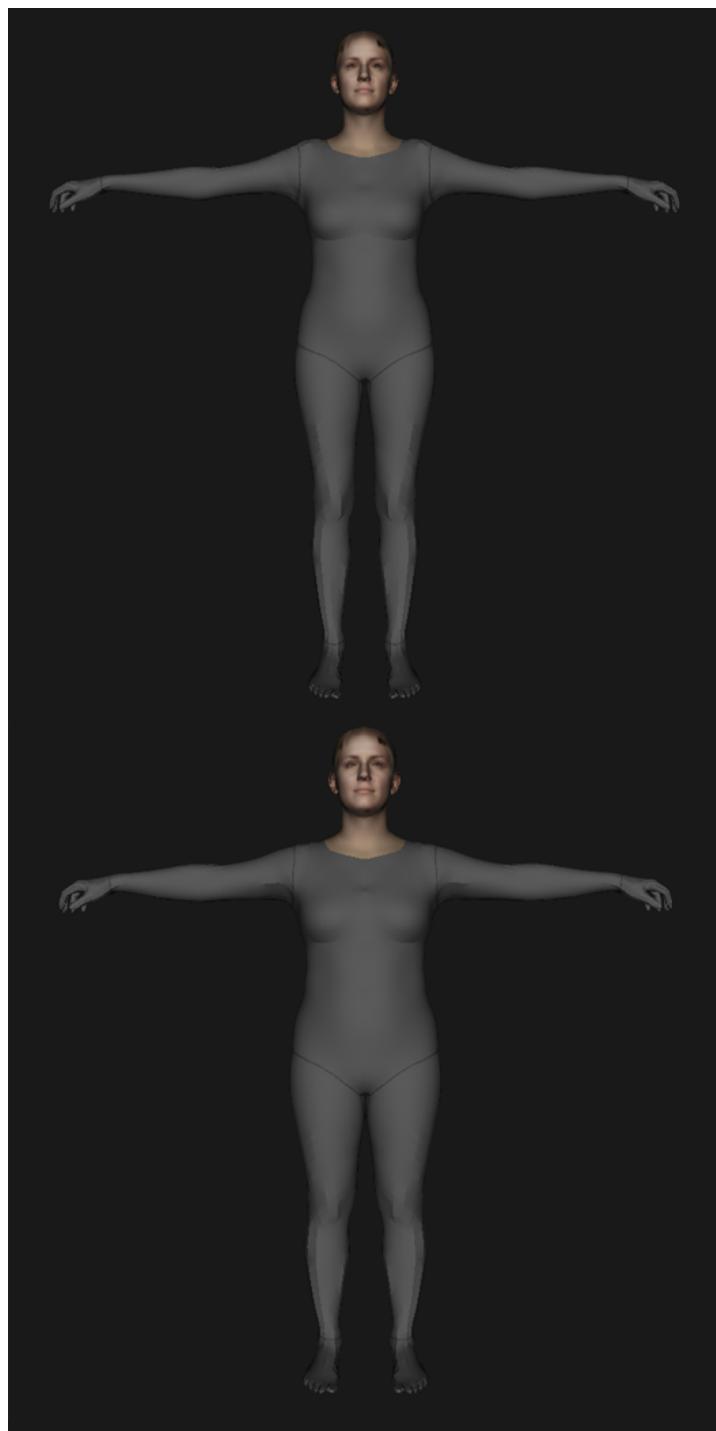


Figura 4.8: Resultados de topología en sujeto número 4.

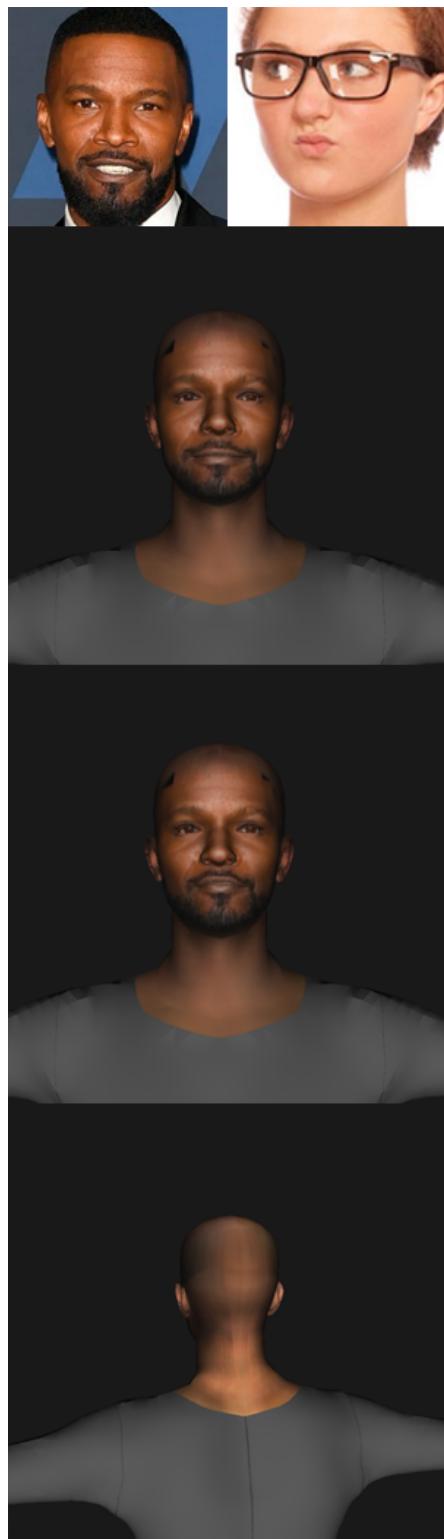


Figura 4.9: Resultados de texturizado y expresión en sujeto número 5.

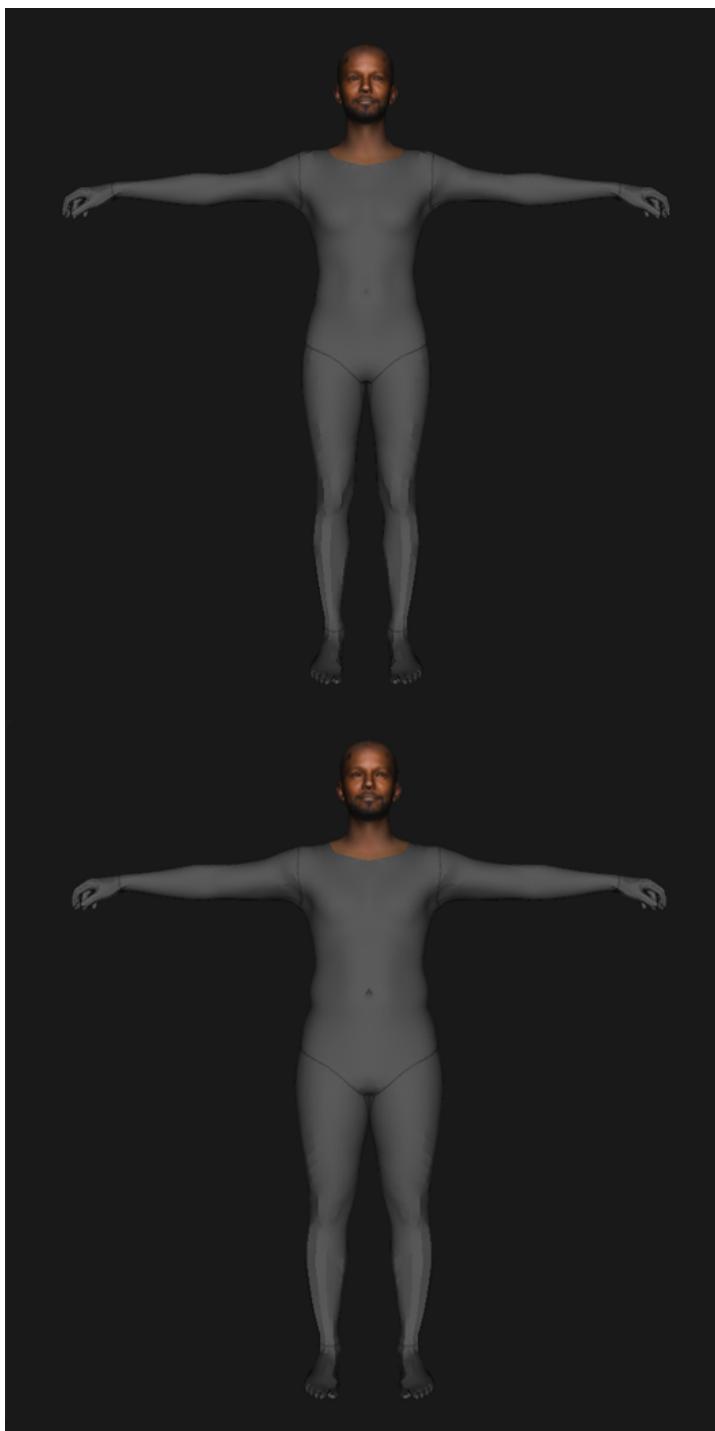


Figura 4.10: Resultados de topología en sujeto número 5.

## 4.2. Alineación de ojos

A continuación se muestra una serie de resultados sobre una gran variedad de sujetos en el proceso de alineación de ojos mediante la red neuronal diseñada. En las imágenes se sigue un orden específico por cada sujeto que, de izquierda a derecha, muestra la imagen de entrada, el modelo como resultado de aplicar la máscara de la Figura 3.55 y el modelo después de corregir la posición del ojo mediante el uso de la red neuronal.

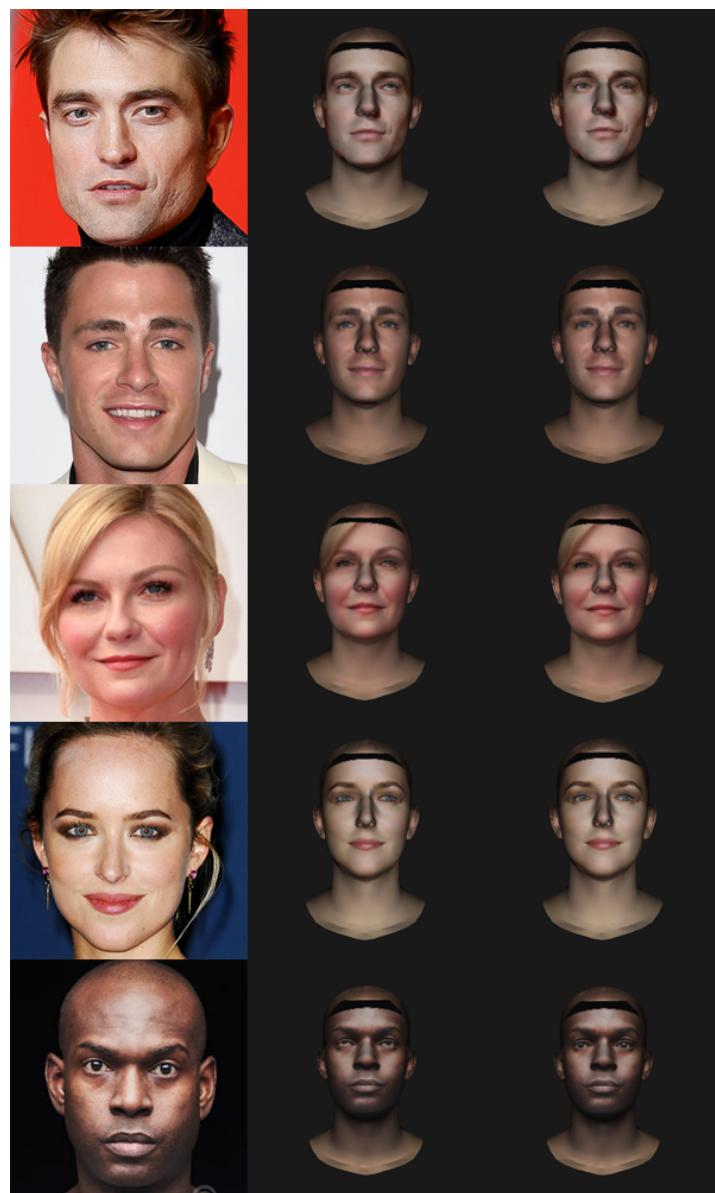


Figura 4.11: Resultados sobre varios sujetos en la utilización de la herramienta de alineación de ojos.

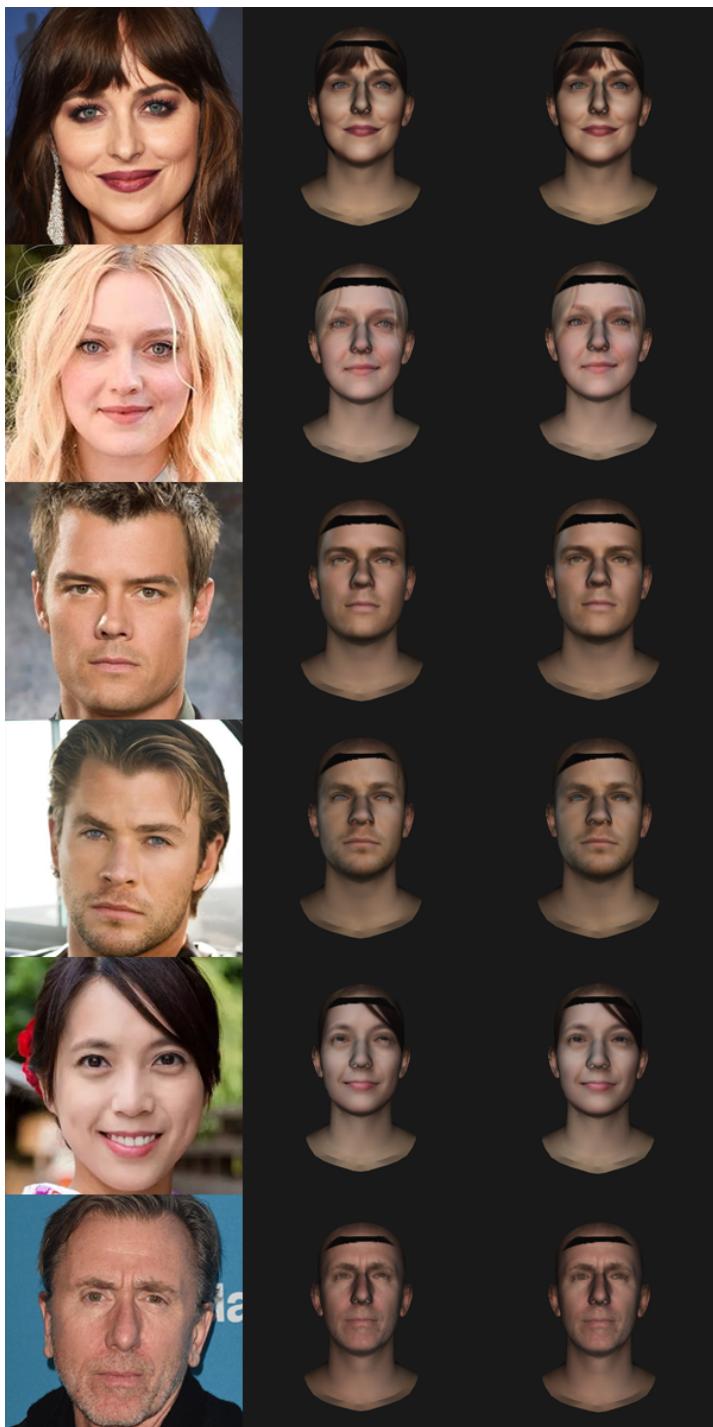


Figura 4.12: Resultados sobre varios sujetos en la utilización de la herramienta de alineación de ojos.

### 4.3. *Training* y optimización

A continuación, se exponen las gráficas de las funciones de pérdida de los diferentes procesos de optimización y aprendizaje.

En primer lugar, se muestra en las Figuras 4.13 y 4.14 las gráficas de la función de pérdida del proceso de optimización en la inferencia de cuerpo. Como se puede observar, las gráficas son diferentes ya que pertenecen a dos sujetos completamente distintos. Sin embargo, se puede observar su tendencia descendente, que se extrae para los demás sujetos.

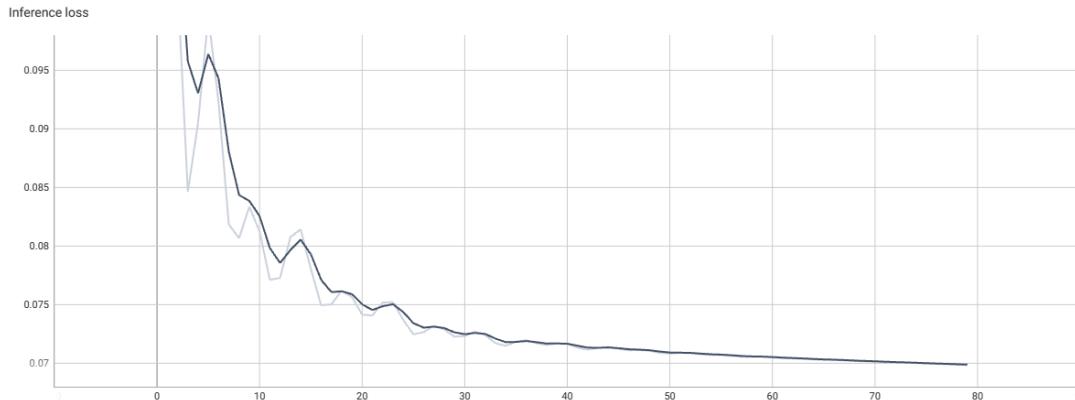


Figura 4.13: Función de pérdida en el proceso de inferencia de cuerpo. Ejemplo 1.

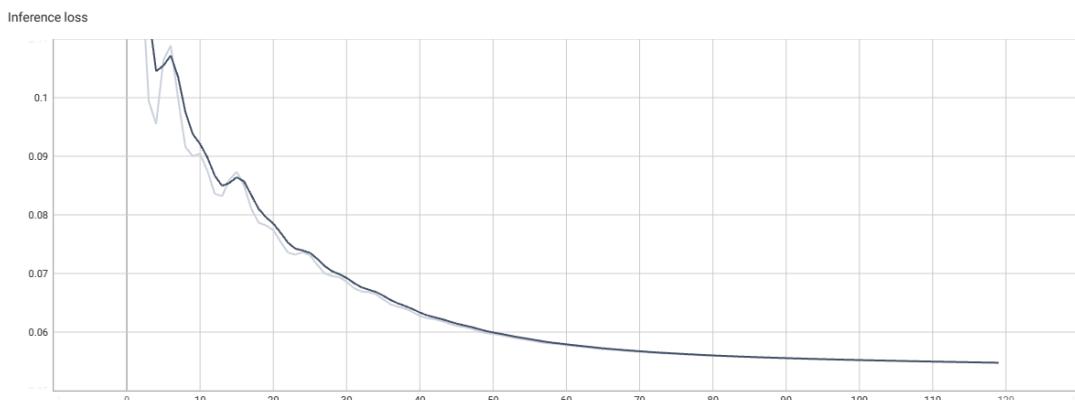


Figura 4.14: Función de pérdida en el proceso de inferencia de cuerpo. Ejemplo 2.

## Capítulo 4. Resultados

---

En segundo lugar, en las Figuras 4.15 y 4.16 se muestran los resultados del entrenamiento de la red neuronal alineadora de ojos sobre los conjuntos de datos de entrenamiento y validación.

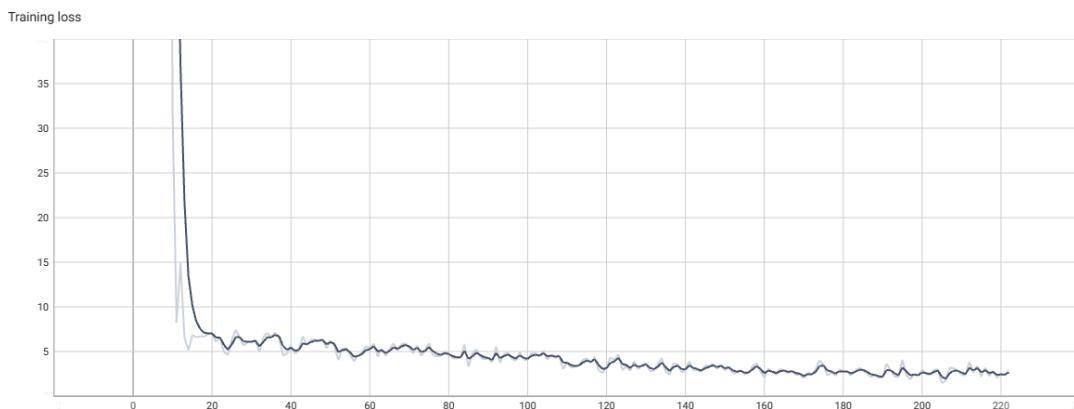


Figura 4.15: Función de pérdida sobre conjuntos de datos de entrenamiento para red neuronal alineadora de ojos.

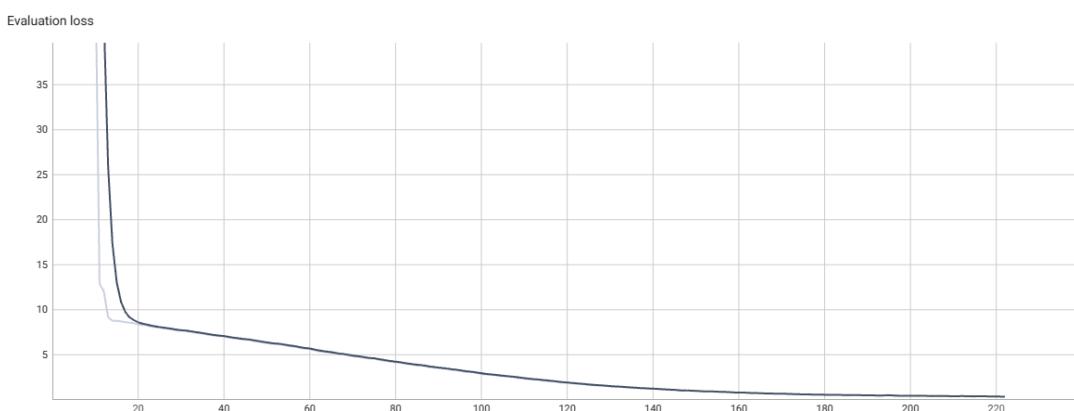


Figura 4.16: Función de pérdida sobre conjuntos de datos de validación para red neuronal alineadora de ojos.

## Capítulo 4. Resultados

---

Por último, a pesar de que la estimación inicial sobre el alineamiento de cabezas es bueno, se experimentan mejoras gracias al optimizador implementado. Sus resultados se pueden observar en la Figura 4.17 para un sujeto arbitrario.

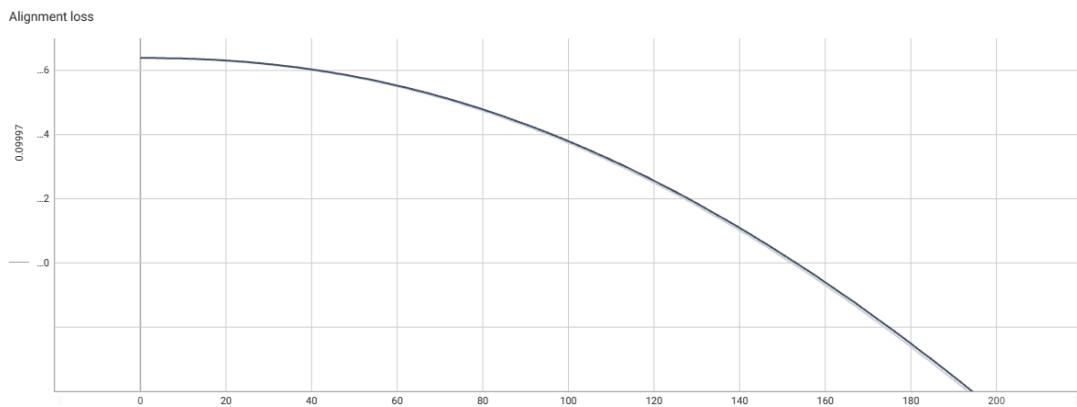


Figura 4.17: Función de pérdida en el proceso de alineación de cabezas.



# 5

## Conclusiones y trabajos futuros

### Conclusiones

En el presente trabajo se logra existosamente la unión de los modelos DECA y SMPL-X en un único modelo integrado. Esto permite combinar las capacidades de reconstrucción y generación de bustos realistas proporcionada por DECA con la capacidad de animación y deformación de cuerpos de SMPL-X mediante una abstracción de datos complejos, permitiendo ser usado por una gran variedad de usuarios.

De esta manera, una vez se tienen los datos de entrada definidos, se es capaz de generar un cuerpo en a penas 1 minuto listo para ser usado en un entorno digital, con mejoras respecto a los modelos originales, y generando unas texturas más consistentes.

Por último, a nivel personal, me ha ayudado enormemente en dos aspectos. En cuanto al mundo de la investigación, me ha servido para entender cómo funciona mejor este campo, relacionandome con él con diversas lecturas de *papers*, estudio de códigos profesionales y todo lo que conlleva crear los entornos virtuales para poder ejecutar los distintos modelos. En cuanto al mundo del Machine Learning, me ha servido para introducirme en una disciplina que me parece muy atractiva, en la cual me he centrado en la etapa final del trabajo donde he tenido que explorar diferentes arquitecturas e implementar cambios en los procesos de entrenamiento.

## Trabajos futuros

A pesar de los logros alcanzados en este trabajo, aún existen áreas que podrían ser exploradas en futuras investigaciones. Algunos posibles trabajos podrían ser:

- **Mejoras en la eficiencia computacional:** Aunque este ha sido un problema que ha sido abordado en todo el proceso, el modelo tarda una media de 90 segundos en ejecutarse al completo en la maquina del autor. Aunque estos tiempos se deben en su mayoría al desempeño intrínseco de DECA, se pueden explorar alternativas para mejorar en este aspecto.
- **Mejoras en la calidad de la texturación:** En el caso específico de DECA, se trabaja con imágenes de baja resolución, lo que resulta en modelos texturizados que podrían beneficiarse de mejoras. Una posible mejora consiste en modificar DECA para trabajar con imágenes de mayor resolución o emplear técnicas de (*super sampling*). Asimismo, el proceso de texturizado puede ser resaltado mediante técnicas de inpainting más sofisticadas para las zonas que presenten ruido.
- **Texturización de cuerpo:** Los límites de la texturización son evidentes, ya que actualmente no existe un método automático para texturizar el cuerpo. En este sentido, se debe considerar la combinación de este trabajo con otros enfoques que sean capaces de generar texturas para el modelo SMPL, lo cual es un área de investigación prometedora para el futuro.
- **Mejora de corrección de ojos:** Aunque el método utilizado en este trabajo puede resultar útil en la mayoría de los casos y ofrecer resultados satisfactorios en una etapa inicial, se evidenciarán sus limitaciones al animar la cara, como por ejemplo en una expresión de sorpresa donde los párpados se abren ampliamente. Para abordar esta cuestión, se propone el uso de una arquitectura UNET[29] para modificar la textura en la zona de los ojos en lugar de realizar una simple traslación, lo cual permitiría obtener mejores resultados.  
Además, se pueden explorar cambios en el proceso de renderizado inverso, como por ejemplo, generar una textura del globo ocular sin incluir la piel circundante.



# 6

## Bibliografía

# Bibliografía

- [1] Body visualizer. <https://github.com/mkocabas/body-model-visualizer>. Abril 2023.
- [2] Correspondencias flame-smplx. <https://github.com/vchoutas/smplx#mano-and-flame-correspondences>. Marzo 2023.
- [3] Flame. <https://flame.is.tue.mpg.de/login.php>. Marzo 2023.
- [4] Inpainting opencv. [https://docs.opencv.org/3.4/d7/d8b/group\\_photo\\_inpaint.html](https://docs.opencv.org/3.4/d7/d8b/group_photo_inpaint.html). Mayo 2023.
- [5] Inpainting technique. [https://www.researchgate.net/publication/238183352\\_An\\_Image\\_Inpainting\\_Technique\\_Based\\_on\\_the\\_Fast\\_Marching\\_Method](https://www.researchgate.net/publication/238183352_An_Image_Inpainting_Technique_Based_on_the_Fast_Marching_Method). Mayo 2023.
- [6] Joblib. <https://joblib.readthedocs.io/en/stable/>. Abril 2023.
- [7] Meta human. <https://www.unrealengine.com/en-US/metahuman>. Junio 2023.
- [8] Opencv. <https://opencv.org/>. Mayo 2023.
- [9] Proyecto flame2smpl-x. <https://github.com/CvHadesSun/FLame2SMPLX>. Abril 2023.
- [10] Seddi. <https://seddi.com>. Julio 2023.
- [11] Unreal engine. <https://www.unrealengine.com/es-ES>. Junio 2023.
- [12] Victoria Fernández Abrevaya, Adnane Boukhayma, Philip HS Torr, and Edmond Boyer. Cross-modal deep face normals with deactivable skip connections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4979–4989, 2020.
- [13] Oswald Aldrian and William AP Smith. Inverse rendering of faces with a 3d morphable model. *IEEE transactions on pattern analysis and machine intelligence*, 35(5):1080–1093, 2012.

## BIBLIOGRAFÍA

---

- [14] Brett Allen, Brian Curless, and Zoran Popović. The space of human body shapes: reconstruction and parameterization from range scans. *ACM transactions on graphics (TOG)*, 22(3):587–594, 2003.
- [15] Brett Allen, Brian Curless, Zoran Popović, and Aaron Hertzmann. Learning a correlated model of identity and pose-dependent body shape variation for real-time synthesis. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 147–156. Citeseer, 2006.
- [16] Brian Amberg, Reinhard Knothe, and Thomas Vetter. Expression invariant 3d face recognition with a morphable model. In *2008 8th IEEE International Conference on Automatic Face & Gesture Recognition*, pages 1–6. IEEE, 2008.
- [17] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. Scape: shape completion and animation of people. In *ACM SIGGRAPH 2005 Papers*, pages 408–416. 2005.
- [18] Thabo Beeler, Bernd Bickel, Paul Beardsley, Bob Sumner, and Markus Gross. High-quality single-shot capture of facial geometry. In *ACM SIGGRAPH 2010 papers*, pages 1–9. 2010.
- [19] Yao Feng, Haiwen Feng, Michael J Black, and Timo Bolkart. Learning an animatable detailed 3d face model from in-the-wild images. *ACM Transactions on Graphics (ToG)*, 40(4):1–13, 2021.
- [20] Thomas Gerig, Andreas Morel-Forster, Clemens Blumer, Bernhard Egger, Marcel Luthi, Sandro Schönborn, and Thomas Vetter. Morphable face models—an open framework. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pages 75–82. IEEE, 2018.
- [21] Nils Hasler, Carsten Stoll, Martin Sunkel, Bodo Rosenhahn, and H-P Seidel. A statistical model of human pose and body shape. In *Computer graphics forum*, volume 28, pages 337–346. Wiley Online Library, 2009.
- [22] Ladislav Kavan and Jiří Žára. Spherical blend skinning: a real-time deformation of articulated models. In *Proceedings of the 2005 symposium on Interactive 3D graphics and games*, pages 9–16, 2005.
- [23] Tianye Li, Timo Bolkart, Michael J Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4d scans. *ACM Trans. Graph.*, 36(6):194–1, 2017.
- [24] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):1–16, 2015.

## BIBLIOGRAFÍA

---

- [25] Ahmed AA Osman, Timo Bolkart, and Michael J Black. Star: Sparse trained articulated human body regressor. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pages 598–613. Springer, 2020.
- [26] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed AA Osman, Dimitrios Tzionas, and Michael J Black. Expressive body capture: 3d hands, face, and body from a single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10975–10985, 2019.
- [27] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572, 1901.
- [28] Kathleen M Robinette, Hans Daanen, and Eric Paquet. The caesar project: a 3-d surface anthropometry survey. In *Second international conference on 3-D digital imaging and modeling (cat. No. PR00062)*, pages 380–386. IEEE, 1999.
- [29] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5–9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.
- [30] Alexandru Telea. An image inpainting technique based on the fast marching method. *Journal of graphics tools*, 9(1):23–34, 2004.

